

EistCockpit

Studienarbeit

Abteilung Informatik

Hochschule für Technik Rapperswil

Frühjahrssemester 2012

Autoren:	David Schöttl, Diego Steiner, Remo Waltenspül
Betreuer:	Thomas Corbat
Projektpartner:	Schweizer Armee, Ristl Bat 20
Experte:	Prof. Dr. M. Stolze
Gegenleser:	-



1.1 Dokumenteninformation

Datum	Version	Änderung	Autor
07.06.2012	1.0	Erste Version des Dokuments	rwaltens
08.06.2012	1.1	Verzeichnisse hinzugefügt	dsteiner
08.06.2012	1.2	Querverweise richtig referenziert	rwaltens
08.06.2012	1.3	Abbildungen & Tabellen beschriftet	dsteiner
08.06.2012	1.5	Rechtschreibfehler korrigiert	dschöttl
08.06.2012	1.6	Abschliessendes Review	alle



1.2 Inhaltverzeichnis





1.3 Abbildungsverzeichnis







1.4 Tabellenverzeichnis





EistCockpit

I. Abstract

David Schöttl, Remo Waltenspül & Diego Steiner



1.5 Dokumenteninformation

Datum	Version	Änderung	Autor
31.05.2012	1.0	Erste Version des Dokuments	rwaltens
04.06.2012	1.1	Dokument überarbeitet	rwaltens



1.6 Abstract

Die Einsatzstelle Telematik (Eist Tm), welche einen Teilbereich eines Richtstrahlbataillons darstellt, hat die Verantwortung über die Planung und die Verwaltung des integrierten militärischen Fernmeldesystem-Netzes (IMFS). Die Eist Tm ist die Anlaufstelle für alle Richtstrahl-Standorte, deshalb werden regelmässig unterschiedliche Statusmeldungen empfangen. Gegenwärtig basiert die Verwaltung all dieser Informationen hauptsächlich auf Meldezetteln in Papierform. Das bisher eingesetzte System ist ineffizient und schwer überschaubar. Zudem haben Ansätze zur Digitalisierung der Informationen nicht oder nur schlecht funktioniert.

Innerhalb dieser Studienarbeit wurde eine Applikation entwickelt, um die Meldungen digital aufzubereiten, zu priorisieren und auf der Empfängerseite, anhand von verschiedenen Kriterien, zu filtern. Während der ersten Projektphase wurden die Anforderungen im Laufe eines vierwöchigen militärischen Dienstes direkt vor Ort mittels Benutzerbefragungen aufgenommen. Dadurch konnte die Benutzeroberfläche ideal auf die Benutzerbedürfnisse abgestimmt werden.

Es wurde ein verteiltes Informations-, Status- und Meldesystem konzipiert. Bei diesem System greifen die Clients über einen RESTful WCF Webservice, welcher die Datenobjekte per JSON serialisiert, auf eine SQLite Datenbank zu.

Damit die Anwendung für zukünftige Anforderungen einfach erweiterbar ist, wurde eine dynamische Plugin-Architektur mittels Microsoft Extensibility Framework (MEF) implementiert.

Die Anwendung wird im Rahmen der Studienarbeit nicht bis zur Produktreife entwickelt, sondern in der nachfolgenden Bachelorarbeit vervollständigt.



EistCockpit

II. Management Summary

David Schöttl, Remo Waltenspül & Diego Steiner



1.7 Dokumenteninformation

Datum	Version	Änderung	Autor
04.06.2012	1.0	Erste Version des Dokuments	rwaltens
04.06.2012	1.1	Dokument überarbeitet	dsteiner
04.06.2012	1.2	Review	dsteiner



1.8 Management Summary

Innerhalb des Stabs eines Richtstrahlbataillons (Ristl Bat) in der Schweizer Armee existiert ein Teilbereich, die Einsatzstelle Telematik (Eist Tm). Deren Verantwortung ist die Planung und das Management des integrierten militärischen Fernmeldesystem-Netzes (IMFS), welches aus Richtstrahlverbindungen (Ristl Vrb), Richtstrahlknoten (Ristl Kn) und diversen Funkgeräten besteht.

Die Eist Tm ist die primäre Anlaufstelle bezüglich diverser Belange wie z.B. technische Probleme, Versorgungsanfragen, Planungsänderungen etc. Regelmässig werden von der Triage neue Statusmeldungen entgegengenommen, welche anschliessend an die Eist Tm weitergeleitet werden. Zurzeit werden diese Informationen hauptsächlich auf Meldezetteln (siehe Abbildung 1 Meldezettel) in Papierform verwaltet. Das gegenwärtige Vorgehen führt zu diversen Problemen, zudem ist es ineffizient und schwer überschaubar. Während dem täglichen Betrieb bestehen die Bedürfnisse Meldungen schnell zu sichten und anhand von bestimmten Kriterien zu filtern, dies kann jedoch mittels des bestehenden Prinzips nicht durchgeführt werden.

Das primäre Ziel lag dabei bei der Konzeption eines Systems zum Erfassen von Statusmeldungen, sowie der Möglichkeit Nachrichten anhand von Kriterien zu filtern. Die Informationen zu allen Standorten sollten übersichtlich dargestellt werden, sowie neue Standorte erfasst werden können. Eine weitere Anforderung war, dass die Applikation für allfällige zukünftige Erweiterungen einfach ausgebaut werden kann.

Es wurde bereits in vergangen Jahren eine Softwarelösung eingesetzt, die jedoch nach einem Systemwechsel nicht mehr kompatibel war. Seit diesem Zeitpunkt wurden die Daten bzw. Statusmeldungen in einer Excel Tabelle gespeichert.

Durch die Applikation EistCockpit werden bestehende Arbeitsabläufe optimiert. Dies manifestiert sich in verschiedenen Arbeitsprozessen, wie z.B. bei der Erfassung von Meldungen, welche beinahe ohne Zeitverlust direkt dem Empfänger übermittelt werden können. Ausserdem werden Standortinformationen zu einer bestimmten Übung übersichtlich dargestellt, dies ermöglicht ein akkurates und umgehendes Handeln. Zudem fördert es den Überblick über den Verlauf von Übungen.



EistCockpit

III. Extended Management Summary

David Schöttl, Remo Waltenspül & Diego Steiner



1.9 Dokumenteninformation

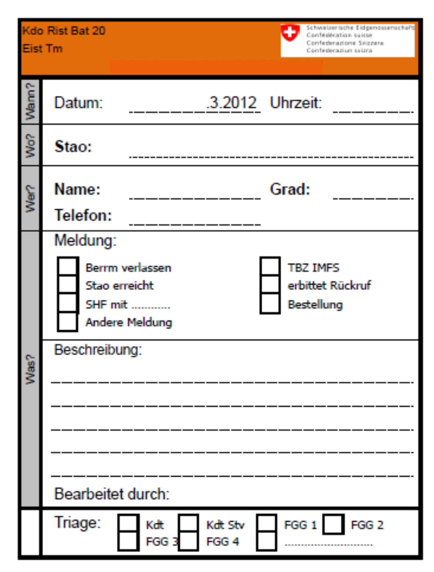
Datum	Version	Änderung	Autor
31.06.2012	1.0	Erste Version des Dokuments	rwaltens
04.06.2012	1.1	Dokument überarbeitet	rwaltens
05.06.2012	1.2	Review	dschöttl



1.10 Ausgangslage

Innerhalb des Stabs eines Richtstrahlbataillons (Ristl
ein Teilbereich, die Einsatzstelle Telematik (Eist Tm).
und das Management des integrierten militärische
welches aus Richtstrahlverbindungen (Ristl Vrb), Ricl
Funkgeräten besteht.

Die Eist Tm ist die primäre Anlaufstelle bezüglich c
Probleme, Versorgungsanfragen, Planungsänderung
Triage neue Statusmeldungen entgegengenommen, v
weitergeleitet werden. Zurzeit werden diese Informati
(siehe Abbildung 1 Meldezettel) in Papierform verwal
zu diversen Problemen, zudem ist es ineffizient und
täglichen Betrieb bestehen die Bedürfnisse Meldunge
bestimmten Kriterien zu filtern, dies kann jedoch mi
durchgeführt werden.



Kdo Rist Bat 20
Eist Tm

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Wann? Datum: ______3.2012 Uhrzeit: ______

Wo? Stao: ______

Wer? Name: ______ Grad: ______
Telefon: ______

Meldung:
- Berrm verlassen
- Stao erreicht
- SHF mit ....
- Andere Meldung
- TBZ IMFS
- erbittet Rückruf
- Bestellung

Was? Beschreibung:
______
______
______
______
______

Bearbeitet durch:

Triage: Kdt, FGG 3, Kdt Stv, FGG 4, FGG 1, FGG 2, ...

Abbildung 1 Meldezettel

Aufgrund von den festgestellten Missständen wurde im Projektteam beschlossen, dass man eine Anwendung entwickeln möchte, die die Abläufe vereinfacht und die Effizienz massgeblich verbessert.

Das primäre Ziel lag dabei bei der Konzeption eines Systems zum Erfassen von Statusmeldungen, sowie der Möglichkeit Nachrichten anhand von Kriterien zu filtern. Die Informationen zu allen Standorten sollten übersichtlich dargestellt werden, sowie neue Standorte erfasst werden können. Eine weitere Anforderung war, dass die Applikation für allfällige zukünftige Erweiterungen einfach ausgebaut werden kann.

Es wurde bereits in vergangen Jahren eine Softwarelösung eingesetzt, die jedoch nach einem Systemwechsel nicht mehr kompatibel war. Seit diesem Zeitpunkt wurden die Daten bzw. Statusmeldungen in einer Excel Tabelle gespeichert.



1.11 Vorgehen & Technologien

Um die Anforderungen an das zu entwickelnde System zu eruieren, nahmen alle Projektmitglieder an einem vierwöchigen Dienst der Schweizer Armee teil. Dabei wurden Benutzerbefragungen durchgeführt um die Bedürfnisse der potentiellen Benutzer zu erkennen. Die gewonnen Erkenntnisse wurden in einem weiteren Schritt analysiert. Aus diesen Analysen wurden drei verschiedene generische Benutzergruppen mit bestimmten Charakteristiken (Personas) geschaffen. Die Personas wurden in der Entwurfsphase rege genutzt, da es ein zentrales Anliegen war, dass die Handhabung sowie die Oberfläche ideal auf die Benutzertypen zugeschnitten sind.

Während des Wiederholungskurses wurde das Projekt dem höheren Kader vorgestellt. Die anschliessende Diskussionsrunde brachte diverse neue Aspekte und Inputs, welche wiederum in der weiteren Entwurfsphase miteinbezogen wurden.

Die Erfahrungen, Einblicke die im Laufe des WK gesammelt wurden, waren äusserst hilfreich für das Gestalten der Benutzeroberfläche sowie der Konzeption des Systems. Nach dem Absolvieren der Dienstleistung beim Richtstrahlbataillon 20 wurden alle gewonnen Daten ausgewertet. Anhand der gesammelten Arbeitsresultate, Artefakte, Benutzerbefragungen etc. wurde ein erster Entwurf für die Benutzeroberfläche erstellt. Dieser Grobentwurf wurde anschliessend im Team ausführlich besprochen und auf allfällige Defizite hinsichtlich Benutzbarkeit geprüft.

Die erkannten Mängel der ersten Skizze wurden beim Papier-Prototyp behoben. Der Papier-Prototyp wurde darauf mit einem Benutzer aus der Triage sowie einem ehemaligen Benutzer auf die Bedienbarkeit getestet. Diese Tests lieferten wertvolle Erkenntnisse über Schwierigkeiten bei der Erreichung von vorgegebenen Aufgaben. Schlussendlich überarbeitete man die Oberfläche, ausgehend von den gewonnen Beobachtungen.

Parallel wurden erste Prototypen entwickelt um Sicherzustellen, dass grundlegende Architekturentscheide umgesetzt werden können. Es wurde ein Prototyp entwickelt, welcher eine zur Laufzeit erweiterbare Benutzeroberfläche implementiert, die mittels des Microsoft Extensibility Framework (MEF) realisiert wurde. Die Clients greifen über einen RESTful WCF Webservice, auf eine SQLite Datenbank zu. Dieser Webservice sowie der Zugriff wurden in eigenen Prototypen realisiert.

Nach den erfolgreichen Prototypen wurde die Projektstruktur vorbereitet sowie die gesamte Datenbank mittels ADO .NET Mapping aufgesetzt. Im Team wurde dabei entschieden, dass die Applikation mit .NET C# implementiert wird, dies hauptsächlich aufgrund des vorinstallierten .NET Frameworks. Die Benutzeroberfläche wird mit der deskriptiven Beschreibungssprache Extensible Application Markup Language (XAML) entworfen.



1.12 Ergebnisse

1.12.1 Arbeitsergebnisse

Trotz verschiedenen kleineren Hürden wurde ein stabiles lauffähiges System entwickelt, welches auf einer ausbaufähigen Plugin Architektur aufsetzt. Dabei wurden bereits die zwei Erweiterungen (Plugins) für die Anzeige bzw. das Erfassen von Meldungen, sowie der Verwaltung von Standorten zu einem ausgewählten Kontext implementiert. Wobei der Kontext eine Liste mit erfassten Diensten und dazugehörigen Übungen darstellt. Der Zugriff von den Clients wird über einen RESTful Webservice durchgeführt. Folge dessen wäre es auch möglich mit weiteren Geräten auf die Objekte zuzugreifen. Dies stellt jedoch gegenwärtig keine Option dar, da von Seiten des Auftraggebers gefordert war, dass keine zusätzliche Hardware verwendet wird.

Die Anwendung wurde mit dem .NET Framework 4.0 entwickelt. Es handelt sich um eine Client/Server Architektur, bei der über einen RESTful WCF Webservice auf die per JSON serialisierte Objekte zugegriffen wird. Dabei werden die Daten serverseitig in einer SQLite Datenbank persistiert. Die Datenbank umfasst bereits einige Tabellen, welche zurzeit mit den bestehenden Plugins noch nicht benötigt werden. Diese können jedoch für zukünftige Erweiterungen benutzt werden.

Es konnten nicht alle gewünschten Features realisiert werden, da der Aufwand zum Teil unterschätzt wurde. Diese werden jedoch im Backlog aufgelistet und können in einer Folgearbeit implementiert werden.

1.12.2 Erkenntnisse

Für das Team war die Studienarbeit eine wertvolle Erfahrung, da bisherige Kenntnisse im Umgang mit :NET C# weiter vertieft wurden. Ferner konnte das Wissen aus dem Modul Software Engineering 3 über das Projekt-Vorgehensmodel Scrum in einem eigenen Projekt angewendet werden. Zum Teil herrschte jedoch ein bisschen Unsicherheit, in Bezug auf welche Artefakte erstellt werden müssen, und wie man genau nach Scrum vorgehen soll. Aus diesem Grund haben wir einschlägige Fachliteratur über Scrum[1] konsultiert.

Abschliessend kann diese Studienarbeit sicherlich als Erfolg gewertet werden, obgleich nicht alle Vorstellungen und Erwartungen, die man beim Beginn des Projekts hatte, vollumfänglich umgesetzt werden konnten. Mit dem Beenden der Studienarbeit steht jedoch nun eine solide Grundlage für eine Folgearbeit.

[1] Gloger, Boris: Scrum Produkte zuverlässig und schnell entwickeln Auflage 1 2008



1.13 Ausblick

Das Projekt EistCockpit befindet sich noch in der Entwicklungsphase und kann daher erst als Prototyp betrachtet werden. Damit die Applikation im Betrieb auch wirklich eingesetzt werden kann, sind noch einige Funktionen zu implementieren. Die Grundlagen sind vorhanden, in einem nächsten Schritt müsste man jedoch weitere Plugins entwickeln wie z.B. ein Bereitschafts-, Netzplan-, Vereinsverwaltung-, und Gefechtsjournalplugin. Des Weiteren müssten die bestehenden zwei Plugins erweitert werden.

Das bestehende Projekt EistCockpit wird in der folgenden Bachelorarbeit weitergeführt. Ziel ist es, dass nach dem Abschluss der Bachelor Thesis eine einsatztaugliche Anwendung besteht, die sich nahtlos in die Abläufe der Eist Tm integriert.



EistCockpit

IV. Technischer Bericht

David Schöttl, Remo Waltenspül & Diego Steiner



1.14 Dokumenteninformation

Datum	Version	Änderung	Autor
07.06.2012	1.0	Erste Version des Dokuments	rwaltens
08.06.2012	1.1	Abschnitt Developer Manual hinzugefügt	rwaltens



1.15 Einleitung

Beim ersten Dokument handelt es sich um das Visionsdokument (siehe Abschnitt 2 Vision), welches den Grund und die Visionen für die Entwicklung von EistCockpit aufzeigt. In diesem Dokument werden Konkurrenzprodukte und die Zielgruppe analysiert.

Das nächste Dokument Projektmanagement (siehe Abschnitt 3 Projektmanagement) gibt einen Überblick wie das Projekt geplant wurde, des Weiteren werden die Risiken aufgelistet sowie das Vorgehensmodel dokumentiert.

Das Kapitel Vorstudie (siehe Abschnitt 4 Vorstudie) behandelt die Analyse der aktuellen Situation sowie der Aufnahme von Bedürfnissen einzelner Benutzer mittels Benutzerbefragungen. Diese Interviews wurden als Grundlage für den Entwurf der Personas verwendet.

Im folgenden Kapitel Anforderungen (siehe Abschnitt 5 Anforderungen) werden hauptsächlich die nicht funktionalen sowie die funktionalen Anforderungen definiert. Weiter werden die verwendeten Tools aufgelistet und kurz die Design Constraints für das Projekt dokumentiert.

Das Kapitel Domain Analyse (siehe Abschnitt 6 Domain Analyse) geht auf die Prozesse ein, die für das Projekt relevant sind. Anschliessend an die Abläufe wird das Domain Model erläutert, welches aus den Vorstudien hervorgeht. Danach wird die Entwicklung der Benutzeroberfläche über mehrere Testzyklen und das Redesign genau dokumentiert.

Das Kapitel Entwurf (siehe Abschnitt 7 Entwurf) widmet sich in einem ersten Teil den wichtigsten Design Entscheiden. In einem weiteren Schritt wird die Architektur beschrieben. Abschliessend werden in diesem Dokument die benutzten Patterns nochmals aufgeführt und kurz vorgestellt.

Im Kapitel Prototyp (siehe Abschnitt 8 Prototypen) werden alle entwickelten Prototypen eingehend beschrieben, dies umfasst die Architektur sowie gewonnene Erkenntnisse.

Das Kapitel Realisierung und Test (siehe Abschnitt 9 Realisierung & Test) dient in erster Linie dazu, mit System-, Unittests nachzuweisen, dass die geforderten Anforderungen korrekt implementiert wurden. Im letzten Abschnitt werden zudem Screenshots vom externen Design dargestellt.

Das Kapitel Developer Manual (siehe Abschnitt 10 Developer Manual) erklärt wie ein Entwickler vorgehen muss, um an diesem Projekt weiterzuarbeiten.

Abschliessend werden im letzten Kapitel Schlussfolgerung (siehe Abschnitt 11 Schlussfolgerung) die erzielten Ergebnisse und die Weiterentwicklungsmöglichkeiten für ein Folgeprojekt aufgeführt.



EistCockpit

2 Vision

David Schöttl, Remo Waltenspül & Diego Steiner



2.1 Dokumenteninformation

Datum	Version	Änderung	Autor
15.03.2012	1.0	Erste Version des Dokuments	rwaltens
28.05.2012	1.1	Review	dsteiner
30.05.2012	1.2	Anpassung Zweck des Dokuments	dsteiner



2.2 Allgemein

2.2.1 Zweck des Dokumentes

Die Absicht dieses Dokumentes ist es auf einem hohen Abstraktionslevel kurz die Bedürfnisse der zukünftigen Benutzer zu definieren. Es soll auch aufzeigen wieso diese Erfordernisse bestehen. Zudem soll mittels des Visions-Dokuments ersichtlich sein, was der Nutzen bzw. Ziele der Anwendung sein soll und wie sich das System vermutlich wirtschaftlich positionieren könnte.

Es wird nur grob erläutert aus welchen Gründen dieses Projekt ins Leben gerufen wurde und was die Visionen bzw. Ideen sind. Für detaillierte Informationen wird auf weitere Dokumente wie dem Projektplan, Architekturdokument etc. verwiesen. Zurzeit bestehen keine weiteren Projekte die in direktem Zusammenhang mit diesem stehen.

Das Visions-Dokument ist in die sechs Abschnitte Einleitung, Positionierung, Kundenbeschreibung, Produktübersicht, Produkteigenschaften, Dokumentationsanforderungen sowie den Anhang unterteilt. Im ersten Kapitel "Einleitung" wird kurz das Projekt mit dessen Zielen erläutert. Anschliessend folgt das zweite Kapitel "Positionierung", in welchem beschrieben wird, wie die gegenwärtige Marktsituation aussieht und wie gut die Erfolgschancen stehen. Das dritte Kapitel "Interessensgruppen und Benutzerbeschreibung" zeigt die zukünftigen Benutzer sowie ihre Interessen und Verhaltensmerkmale. Danach kommt das vierte Kapitel "Produktübersicht", bei dem die Idee des zu entwickelnden Produkts sowie die Perspektive beschrieben wird. Im Kapitel "Produkteigenschaften" werden die wichtigsten Funktionen der Anwendung erläutert. Abschliessend wird beim Kapitel "Dokumentationsanforderungen" kurz zusammengefasst, was die Anforderungen bezüglich der Dokumentation sind.

2.2.2 Gültigkeitsbereich

Dieses Dokument gilt als Grundlage des Projektes und ist daher über die gesamte Projektdauer gültig (20.02 bis 07.06.2012).

2.2.3 Definitionen und Abkürzungen

Siehe „Glossar“.



2.3 Positionierung

2.3.1 Geschäftschancen

- Die Abläufe können optimiert werden (Effizienzsteigerung) => Führt zu vermindertem Arbeitsaufwand
- Klarer Prozess beim Entgegennehmen von Meldungen => Steigert Motivation von Benutzern/Mitarbeitern
- Meldungen gehen nicht mehr verloren => System verlässlicher
- Einarbeitungszeit für neue Mitarbeiter kürzer aufgrund von hilfreichen Zusatzinformationen

2.3.1.1 Problembeschreibung

Problem	Meldungen gehen verloren, Mitarbeiter wissen nicht wie reagieren auf bestimmte Meldungen, Meldungen sind unvollständig
Beeinflusst wenn	Die Mitarbeiter in der Triage sowie die Empfänger in der Eist TM
Auswirkungen	Empfänger müssen wegen fehlenden Informationen nachfragen, Triage-Mitarbeiter muss Anrufer nochmals kontaktieren
Erfolgreiche Lösung wäre	effizient, benutzerfreundlich, zuverlässig, fehlerfrei, einfach zu bedienen, mit guten Zusatzinformationen versehen

Tabelle 1 Problembeschreibung

2.3.1.2 Produkt Positionsbeschreibung

Die nachfolgende Tabelle zeigt kurz auf wie sich das Produkt ungefähr positionieren könnte und mit welchen Schlüsseleigenschaften es sich gegen bestehende Angebote abheben kann.

Zielgruppe	Grössere Organisationen, welche Meldungen entgegennehmen und effizient weiterleiten möchten.
Nutzen, Chancen	Effizientere, zuverlässigere sowie schnellere Verarbeitung von Meldungen
Produktkategorie	Meldesystem (mit Erweiterungen)
Kaufgründe	Äusserst benutzerfreundlich, kurze Einarbeitungszeit, Effiziente Abwicklung
Konkurrenzprodukte	
Positive Eigenheiten	Bietet Zusatzinformationen für Mitarbeiter in Call Center, Vergangene Meldungen werden in einer History angezeigt (Tracking)

Tabelle 2 Positionsbeschreibung Projekt EistCockpit



2.3.1.3 Interessensvertreter und Benutzerbeschreibungen

Marktdemographie

Für die Analyse der Marktdemographischen Situation wurde die Führungsunterstützungsbrigade 41[2] als grössere Zielgruppe definiert. Wobei bis anhin noch nicht sichergestellt ist, ob das Produkt in der gesamten Brigade eingesetzt wird. In erster Linie wird es in einem Pilotversuch im Richtstrahl Batallion 20 auf die Praxistauglichkeit getestet. Erst nach der erfolgreichen Auswertung und Bewährung der Software wird das System eventuell im grösseren Verbund benutzt.

Wenn man nur das Ristl Bat 20 betrachtet, werden durch die zu entwickelnden Anwendung etwa 35 Personen betroffen sein. Falls das Meldesystem in einem zweiten Schritt in der gesamten Führungsunterstützungsbrigade 41 eingesetzt werden soll, kann man die geschätzte Zahl von 35 involvierten Mitarbeitern auf die komplette Brigade hochrechnen. Dies impliziert bei weiteren sieben Batallions die achtfache Anzahl betroffener Personen. Somit werden mit der Applikation etwa 280 Personen angesprochen.

In nächster Zeit ist nicht mit einer Dezimierung der Einheit zu rechnen, dies kann sich aber in den folgenden 10-20 Jahren massgeblich ändern.

Finanziell wird dieses Projekt nicht unterstützt, jedoch besteht die Möglichkeit, dass einzelne AdA einen Dienst teilweise für die Instandhaltung, Aktualisierung, etc. nutzen können.

Zusammenfassung Interessensvertreter

Höheres Kader

Beschreibung	**Verantwortlichkeiten**
Sie leiten eine Abteilung und haben deshalb mehr Kompetenzen, dies impliziert für die Anwendung eine spezielle Ansicht mit zusätzlichen Informationen	Sie geben verschiedene Inputs, Verbesserungsvorschläge für die Entwicklung

Tabelle 3 Interessensvertreter höheres Kader

Projektleiter

Beschreibung	**Verantwortlichkeiten**
Sie leiten das gesamte Projekt, und haben deshalb eine grosse Verantwortung	Die Projektleiter sind massgeblich bei der Einteilung und Planung des Projekts beteiligt, beobachten den Projektverlauf

Tabelle 4 Interessensvertreter Projektleiter

Führungsunterstützungsbrigade

Beschreibung	**Verantwortlichkeiten**
Bei dieser Brigade handelt es sich um die übergeordnete Einheit, welche bestimmte militärische Aufträge koordiniert	Schaut sich Projektstand von Zeit zu Zeit an, keine grossen Verantwortlichkeiten (da in erster Linie für Batallion)

Tabelle 5 Interessensvertreter Führungsunterstützungsbrigade

---

[2] [url1]: Führungsunterstützungsbrigade 41, *http://de.wikipedia.org/wiki/F%C3%BChrungsunterst%C3%BCtzungsbrigade_41* (15.03.2012)



Soldaten (Mitarbeiter)

Beschreibung	Verantwortlichkeiten
Führt die Befehle aus, welche ihm erteilt werden, Arbeitet in der Triage oder der Eist TM	Hat nicht viel Einblick bei der Entwicklung der Anwendung, Eventuell muss er erste Prototypen testen

Tabelle 6 Interessensvertreter Soldaten

Zusammenfassung Benutzer

Triage Mitarbeiter

Beschreibung	Verantwortlichkeiten	Interessensvertreter
Nimmt Telefone entgegen und erfasst neue Meldungen	Erfasst neue Meldungen, Zeigt vergange Meldungen an	Soldaten (Mitarbeiter)

Tabelle 7 Beschreibung Benutzer

Gruppenführer Triage

Beschreibung	Verantwortlichkeiten	Interessensvertreter
Koordiniert Abläufe, teilt Leute für diverse Schichten ein, nimmt zum Teil selber Telefone entgegen	Meldungen verwalten, Status von Meldungen aktualisieren, Meldungen erfassen	Wird am besten durch Interessensvertreter "Soldaten (Mitarbeiter)" vertreten

Tabelle 8 Beschreibung Gruppenführer Triage

Empfänger

Beschreibung	Verantwortlichkeiten	Interessensvertreter
Der Empfänger kann sich in verschiedenen Abteilungen (bsp. FGG1, FGG2 etc.) befinden. Er nimmt die Meldungen entgegen und leitet, falls es sich nicht nur um eine Information handelt, daraus eine Aufgabe für eine andere Abteilung ab.		Beim Empfänger handelt es sich häufig um das Höhere Kader

Tabelle 9 Beschreibung Empfänger



Benutzerumgebung

Grundsätzlich sind in der Triage während dem Tag immer drei Personen anwesend, welche die Telefone entgegennehmen. In der Nacht existiert nur eine "Telefonwache" bestehend aus einem AdA. Bei den Empfänger handelt es sich in der Regel um die Abteilungsleiter der betroffenen Zellen (Abteilungen), anderseits können auch normale Mitarbeiter (Soldaten) Meldungen entgegennehmen, sofern sie die benötigten Kompetenzen dazu haben.

Die Dauer für einen gesamten Meldungszyklus kann stark variieren, dies ist vor allem abhängig, ob es sich nur um eine Information handelt oder ob daraus ein weiterer Auftrag entsteht. Es ist möglich, dass der Meldeprozess nur ein paar Minuten dauert, oder bei komplexeren Meldungen mehrere Stunden.

Das System wird nur innerhalb von Gebäuden (meist ZSA, Bunker) eingesetzt, eine mobile Verwendung ist zurzeit nicht vorgesehen.

Es existieren weitere Anwendungen auf dem Teplas Server, gegenwärtig ist jedoch keine Anbindung an bestehende Applikationen geplant, mit der Ausnahme vom Active Directory. Damit Personen direkt über eine Auswahlliste ausgewählt werden können, wird das AD des MS Servers integriert.

Profile Interessensvertreter

Höheres Kader

Beschreibung	Erteilt Befehle, Koordiniert die Abläufe und Aufträge, trifft Entscheidungen
Typ	Kader hat durchschnittliches technisches Wissen (Anwenderkenntnisse)
Verantwortlichkeiten	Meldungen entgegennehmen, aus Meldungen neue Aufträge generieren, Status von Übungen beobachten
Erfolgskriterien	Meldungen vollständig an zuständige Personen übermittelt werden, Meldungszyklus effizient abgewickelt wird
Wie Involviert	Versuchspersonen bei ersten Prototypen, Ansprechpersonen bei Requirement Engineering
Lieferbare Ergebnisse	Erstellte Aufträge, Weitergeleitete Meldungen

Tabelle 10 Profil Interessensvertreter höheres Kdaer



Projektleiter

Beschreibung	Der Projektleiter koordiniert das Projekt und versucht die wichtigsten Projektparameter ständig im Auge zu behalten
Typ	Hat grosses technisches Wissen, sowie gute Führungskompetenzen
Verantwortlichkeiten	Den Rahmen für das Projekt klar zu definieren, kontrollieren wie der aktuelle Stand des Projekts aussieht
Erfolgskriterien	Wenn am Schluss des Projekts eine funktionierende anforderungsgerechte Anwendung besteht, sowie alle Qualitätsansprüche erfüllt werden können.
Wie Involviert	Unterstützt, koordiniert das Projekt
Lieferbare Ergebnisse	Projektdokumentationen

Tabelle 11 Profil Interessensvertreter Projektleiter

Führungsunterstützungsbrigade

Dies Wird nicht detaillierter erläutert, da bis anhin noch nicht feststeht, ob die Brigade das System jemals nutzen wird.

Soldaten (Mitarbeiter)

Beschreibung	Die Mitarbeiter nehmen Telefone entgegen und erfassen neue Meldungen
Typ	Die meisten Mitarbeiter haben nur oberflächliche Computerkenntnisse
Verantwortlichkeiten	Meldungen erfassen, Bisherige Meldungen ansehen, Status von Meldungen bearbeiten
Erfolgskriterien	Meldung wird zeitgerecht und vollständig erfasst
Wie Involviert	Testperson bei Feldtests (Prototypen), Aufnahme Ist-Situation (Requirement Engineering)
Lieferbare Ergebnisse	Erfasste Meldungen

Tabelle 12 Profil Interessensvertreter Soldaten



Benutzerprofile

Triage Mitarbeiter

Die Triage Mitarbeiter werden durch die Interessensgruppe der Soldaten (Mitarbeiter) vertreten, deshalb sei auf diese Gruppe verwiesen.

Gruppenführer Triage

Beschreibung	Die Gruppenführer (häufig Wachtmeister) betreuen die Mitarbeiter in der Triage und erstellen Arbeitspläne.
Typ	Häufig haben die Wachtmeister ein marginal höheres Fachwissen als die Soldaten
Verantwortlichkeiten	Schichtpläne erstellen, z.t. selber Meldungen erfassen, Status von Meldungen bearbeiten
Erfolgskriterien	Das Personal in der Triage ist gut informiert, pünktlich bei der Arbeit und die Meldungen werden vollständig erfasst
Wie Involviert	Während Projektlaufzeit kaum involviert, eventuell bei der Analyse der Ist-Situation
Lieferbare Ergebnisse	Meldungen, Berichte über aktuelle Übungen, Arbeitsplan

Tabelle 13 Benutzerprofil Gruppenführer Triage

Empfänger

Beschreibung	Der Empfänger (häufig ein Abteilungsleiter einer Zelle) nimmt die Meldung entgegen, daraus entsteht z.T. ein neuer Auftrag
Typ	I.d.R. gute technische Kenntnisse, versiert im Umgang mit modernen Mitteln
Verantwortlichkeiten	Meldungen lesen, Aufträge erfassen, Meldungsstatus prüfen bzw. bearbeiten, Übungsrapport drucken
Erfolgskriterien	Für den Empfänger ist das System erfolgreich, wenn die Meldungen vollständig und zeitgerecht bei ihm ankommen.
Wie Involviert	Betreut während der Projektdauer das Projekt und gibt Inputs, Verbesserungsvorschläge, testet erste Prototypen
Lieferbare Ergebnisse	Aus Meldungen erstellte Aufträge, korrigierte Meldungen

Tabelle 14 Benutzerprofil Empfänger



Benutzerbedürfnisse

Prioritätsstufen

- Höchst (1) Höchste Priorität
- Hoch (2) Hohe Priorität
- Mittel (3) Mittlere Priorität
- Tief (4) Tiefe Priorität

Bedürfnis	Prio.	Betrifft	Gegenwärtige Lösung	Vorgeschlagene Lösung
Meldungen gehen nicht verloren	1	Mitarbeiter in Triage	Meldungen werden in Excel-Tabelle erfasst	Meldungen werden direkt über die Applikation erfasst
Meldungen vollständig	1	Mitarbeiter in Eist TM und Triage	Nicht geprüft	System macht Validierungen, und stellt sicher, dass alle benötigten Informationen vorhanden sind
Übersicht über eingegangene Anrufe	2	Mitarbeiter in Triage	Existiert nicht	Anhand einer History sollen die letzten Telefonate sichtbar sein
Priorisierung von Meldungen	2	Mitarbeiter in verschiedenen Zellen	Zurzeit nicht möglich	Einer Meldung kann eine bestimmte Priorität zugewiesen werden.
Status von Übungen anzeigen	2	Abteilungsleiter der verschiedenen Zellen	Zurzeit nicht möglich	Die Applikation erlaubt es zu einer laufenden Übung den Status anzuzeigen.
Erreichbare Personen in Anlage anzeigen	2	Mitarbeiter in der Triage	Zurzeit nicht möglich	Mittels dem System sollen alle in der Anlage anwesenden Personen anzeigbar sein
Informationen für die Entgegennahme von Anrufen	2	Mitarbeiter in der Triage	Viele Plakate verteilt im Raum mit Informationen	Der Ablauf ist sequenziell geführt, dass immer die passenden Informationen sichtbar sind

Tabelle 15 Übersicht Benutzerbedürfnisse



2.3.1.4 Produktübersicht

Idee

Ziel ist es ein elektronisches Meldesystem aufzubauen, bei dem neue Meldungen erfasst und bearbeitet werden können. Dadurch sollen verlorene oder unvollständige Meldungen verhindert werden, dies führt direkt zu einer höheren Qualität, gesteigerter Effizienz und höherer Motivation bei den Mitarbeitern. Weiter soll es auch möglich sein alle in der ZSA anwesenden Personen auf einen Blick zu sehen, dadurch können Mitarbeiter in der Triage schneller reagieren und notfalls die Meldungen an den Stellvertreter weiterleiten.

In einem folgenden Schritt könnte man auch eine Meldungshistory integrieren, die die letzten Meldungen übersichtlich darstellt.

Über die Kernfunktionen hinaus können mit der Zeit weitere Features mittels eines Plugin-Systems hinzugefügt werden. Dadurch wird auch sichergestellt, dass die Software einfach erweiterbar ist oder an neue Systeme angebunden werden kann.

Produktperspektive

Bei dem zu entwerfenden System handelt es sich um ein weitgehend autarkes System. Lediglich eine Anbindung an das Active Directory soll realisiert werden, damit erfasste Personen einfach ausgewählt werden können. In einem weiteren Schritt werden dann eventuell weitere Systeme integriert, wie z.B. das Tool zum Anzeigen des Netzplan BPL oder ein Modul zum Erstellen von Wiki Seiten.

Zusammenfassung Produktfähigkeiten

Kundennutzen	Unterstützende Produkteigenschaften
Übersicht über alle aktuellen Meldungen	Meldungen werden solange sie noch nicht geschlossen wurden, übersichtlich dargestellt und können gefiltert werden
Erreichbarkeit von Personen sichtbar	Beim Eintreten oder Verlassen der Anlage wird vom Personal der Erreichbarkeitsstatus angepasst
Mitarbeiter müssen nicht mehr persönlich die Meldung dem Empfänger überbringen	Die Meldung wird im System erfasst und den zuständigen Personen ähnlich wie per Mail übermittelt
Anrufer die bereits mehrmals telefoniert haben, können an vorhergehenden Mitarbeiter weitergeleitet werden.	Mittels einer History zur Übersicht von den Meldungen der letzten Stunde
Empfänger erhält Meldung vollständig und ohne Umwege	Durch Validierungsprozesse der Software wird sichergestellt, dass die Meldung vollständig erfasst wird
Meldung kann an mehrere Adressaten übermittelt werden	Das System erlaubt es, nicht nur einen sondern auch mehrere Empfänger gleichzeitig zu definieren

Tabelle 16 Zusammenfassung Produktfähigkeiten



Annahmen und Abhängigkeiten

- Es wird davon ausgegangen, dass auf dem Teplas Server im Minimum das .Net Framework 4.0 installiert ist.
- Die Anbindung an das Active Directory des Servers möglich ist
- Die Möglichkeit zur Installation von Software besteht auf dem Server

Preis und Kosten

Da es sich bei diesem Produkt um keine kommerzielle Anwendung handelt, fallen auch keine Verpackungskosten etc. an. Die fertige Software sollte im Idealfall einfach auf den Server kopiert werden können, dadurch entfallen auch allfällige Überlegungen betreffend der Distribution der Software.

Lizenz und Installation

Wie im vorhergehenden Kapitel bereits beschrieben wurde, sollte es möglich sein die Applikation ohne Installation in Betrieb nehmen zu können. Da das System vorwiegend für das Militär entwickelt wird, und keine finanzielle Unterstützung zugesprochen wurde, wird auf eine Lizenz verzichtet.

2.3.1.5 Produkteigenschaften

Meldungen erfassen

Die Applikation bietet eine Möglichkeit um eine Meldung mit allen benötigten Werten zu erfassen. Diese Meldung wird dann gespeichert und bei den zuständigen Personen angezeigt.

Übersicht der letzten Meldungen anzeigen

Für die Mitarbeiter in der Triage werden alle Meldungen eines bestimmten Zeitraums angezeigt. Dadurch können Anrufer unter Umständen wieder mit dem gleichen Mitarbeiter verbunden werden.

Erreichbarkeit von Personen

Im System besteht eine Möglichkeit, abzurufen ob eine bestimmte Person in der Anlage anwesend ist. Falls die benötigte Person nicht erreichbar ist, kann dementsprechend deren Stellvertreter anstelle kontaktiert werden.

Rapport drucken

Für das höhere Kader soll es möglich sein, einen Statusbericht ausdrucken zu lassen. Dieser Rapport enthält eine Auflistung der verschiedenen Übungen und deren Fortschritt.

Meldungen ausdrucken

Im Falle eines Stromausfalls muss es möglich sein, die Meldungen auf normalem Papier auszudrucken. Dadurch kann man problemlos wieder auf den konventionellen Weg umsteigen.

Meldungen priorisieren

Beim Erfassen von Meldungen besteht eine Funktion zur Priorisierung von Meldungen, dies ermöglicht die Unterscheidung von Meldungen und deren Hervorhebung.

Filtern von Meldungen

Falls ein Empfänger viele Meldungen erwartet, möchte er mittels eines Filtersystems seine Auswahl nach bestimmten Bedingungen eingrenzen.



Bericht zu bestimmter Operation

Diese Funktion gibt einen Überblick über eine ausgewählte Übung. Das heisst, es werden alle eingetroffenen Meldungen in einem gewissen Zeitraum angezeigt. Dies erlaubt eine Einschätzung des aktuellen Fortschritts einer Übung.

2.3.1.6 Dokumentationsanforderungen

Bedienungsanleitung

Die Bedienungsanleitung soll Schritt für Schritt ganz klar aufzeigen, wie sämtliche Funktionen der Software genutzt werden können. Um die Anwendung auch für technisch weniger versierte Mitarbeiter zugänglich zu machen, werden zur Illustration Screenshots verwendet, die ersichtlich machen wie man ein bestimmtes Ziel erreichen kann.

Online-Hilfe

Zum derzeitigen Zeitpunkt wird keine Onlinedokumentation zur Verfügung gestellt.

Installationsanleitung, Konfiguration, und Read-Me Datei

Es wird eine von der Bedienungsanleitung unabhängige Installationsanleitung erstellt, die klar Auskunft gibt, wie man die Software auf dem Teplas Server installiert. Vorgesehen ist eine einfache Installation, bei der nur bestimmte Dateien auf den Server kopiert werden müssen.



EistCockpit

3 Projektmanagement

David Schöttl, Remo Waltenspül & Diego Steiner



3.1 Dokumenteninformation

Datum	Version	Änderung	Autor
29.05.2012	1.0	Erste Version des Dokuments	dsteiner
04.06.2012	1.1	Entwicklerinformationen von rwaltens hinzugefügt	rwaltens
07.06.2012	1.2	Entwicklerinformationen von dschöttl hinzugefügt	dschöttl
08.06.2012	1.3	Review	rwaltens



3.2 Allgemein

3.2.1 Zweck des Dokumentes

Dieses Dokument beschreibt das Projektmanagement für das EistCockpit. Dazu gehören Vorgehensmodell und die Planung der Abschnitte (Sprints).

3.2.2 Gültigkeitsbereich

Dieses Dokument gilt als Grundlage des Projektes und ist daher über die gesamte Projektdauer gültig (20.02 bis 07.06.2012).

3.2.3 Definitionen und Abkürzungen

Siehe „Glossar“.



3.3 Vorgehensmodell

Das für das EistCockpit bevorzugte Vorgehensmodell orientiert sich stark an Scrum, um eine möglichst unkomplizierte und agile Softwareentwicklung zu ermöglichen.

Die Zeitlichen Umstände verlangen aber nach einem angepassten Modell; Zu Beginn des Projektes ist das gesamte Entwicklerteam (bestehend aus D. Schöttl, R. Waltenspül und Diego Steiner) sowie Betreuer Thomas Corbat zur Leistung des militärischen Dienstes (Wiederholungskurs, WK) verpflichtet. Diese Dienstleistung beansprucht rund 4 Wochen der Projektzeit, in welcher keine Arbeitszeit für Externe Projekte garantiert wird. Der Rahmen des WKs bietet allerdings die besten Voraussetzungen um die Vorstudien durchzuführen, weil der Endbenutzer direkt am Arbeitsplatz beobachtet und interviewt werden kann. Ausserdem sind die Ansprechpersonen auf Seite des Kunden ebenfalls vor Ort, was die Kommunikation in dieser ersten Phase extrem erleichtert. Um diesen Umständen gerecht zu werden wurde das Vorgehensmodell für diesen Zeitabschnitt in an RUP angelehnte Phasen unterteilt. In diesen Phasen wird der Rahmen des Projektes evaluiert, die Prozesse analysiert und die Bedürfnisse der Benutzer und Kunden aufgenommen.

In dem anschliessenden Sprint werden dann die gewonnenen Erkenntnisse ins Backlog übertragen und priorisiert, sodass man sich während den drei darauffolgenden Sprints auf die Entwicklung konzentrieren kann.

3.3.1 Elemente

3.3.1.1 Rollen

Product-Owner	Wird wegen des Platzmangels von David Schöttl besetzt, der Aufgrund seiner Fachkenntnisse und Erfahrung am besten geeignet ist.
Scrum-Master	Übernimmt Diego Steiner, weil auch hier keine bessere Lösung möglich ist.

Tabelle 17 Projektrollen



3.3.1.2 Sitzungen

Sprint Planung	Am Ende des vorherigen Sprints wird der jeweils nächste Sprint mit der Abschätzung der Tickets geplant.
Daily-Scrum Meeting	Am Anfang jedes Arbeitstages werden während 5 Minuten die vergangenen Arbeitspakete sowie die Ziele des Tages besprochen.
Betreuer Sitzung	Im Abstand von ein bis zwei Wochen sitzt das Entwicklungsteam mit dem Betreuer zusammen um ihm den aktuellen Stand zu schildern und Fragen zu stellen.
Sprint Review	Am Ende eines jeden Sprints werden alle Tickets und deren Ausführung von einem anderen Projektmitglied gereviewt und anschliessend im Redmine auf „Closed“ gesetzt (siehe „3.6.1 Review der Tickets“).

Tabelle 18Sitzungstypen

3.3.1.3 Artefakte

Product Backlog	Das Product Backlog wir über eine Unbefristete Version „Backlog“ im Redmine realisiert, in welchem alle unangetasteten Userstories gelagert werden.
Sprint Backlog	Das Sprint Backlog wir ebenfalls über eine Version im Redmine realisiert. Im Unterschied zum Product Backlog ist die Version jedoch auf den Sprint befristet und wird mit diesem geschlossen.
Impediment Log	Die Schwierigkeiten werden auf eine dafür eingerichteten Wiki-Seite im Redmine gelistet. Der Scrum-Master überprüft dieses in periodischen Abständen und regiert entsprechend.
Burndown Charts	Die Burndown Charts werden im Wesentlichen über das Gantt-Diagramm im Redmine realisiert. In speziellen Fällen können aber auch noch einzelne Diagramme erstellt werden.

Tabelle 19 Scrum Artefakten

3.3.2 Prinzipien

Die Studienarbeit soll sich an den nachfolgenden Philosophien orientieren.

DRY

Don´t Repeat Yourself – Das Prinzip des sich-nicht-wiederholens ist wichtig um den Code stets wartbar zu halten. (Hunt & Thomas, 1999)[3]

---

[3] Siehe auch [url2]: Don't repeat yourself, http://en.wikipedia.org/wiki/DRY (08.06.2012)



Projektplan

Das gesamte Projektmanagement für das EistCockpit wird über das Redmine unter https://sinv-56026.edu.hsr.ch/ abgewickelt.

3.3.3 Releases

Da das Vorgehensmodell des Projektes stark an Scrum angelehnt ist (siehe dazu Kapitel „3.5 Vorgehensmodell“) gibt es nur am Ende einen Release.

Zum Abschluss jedes Sprints soll jedoch eine lauffähige Version der Software stehen.

Version	Datum	Beschreibung
1.0	07.06.2012	Finale Version mit Installer

Tabelle 20 Releasebeschreibungen

3.3.4 Phasen und Sprints

Die Projektdauer von 14 Wochen wurde in Phasen und Sprints unterteilt, deren Beendigung jeweils einen Milestone darstellt.



Detailliertere Informationen zu jedem Sprint sind in der Roadmap und auf der einzelnen Version im Redmine zu finden.



Phase „Inception“

Inception

Dauer	19.02. bis 26.02.2012
Ziele	• Infrastuktur bereitstellen • Aufgabenstellung ausarbeiten • Grundstein legen
Artefakte	
Termine	• Sitzung 20.02.2012 • Sitzung 22.02.2012 • Sitzung 24.02.2012

Tabelle 21 Beschreibung Phase Inception

Phase „Elaboration I“

Elaboration I

Dauer	27.02. bis 11.03.2012
Ziele	• Interviews vorbereiten • Interviews durchführen
Artefakte	• Interviewplan • Interviews
Termine	• Situng 05.03.2012 Nachmittags • Situng 05.03.2012 Abends

Tabelle 22 Beschreibung Phase Elaboration 1



Phase „Elaboration II“

Elaboration II

Dauer	12.03. bis 25.03.2012
Ziele	• Personas erstellen • Vision verfassen
Artefakte	• Personas • Vision • Beobachtungsdokument Triage
Termine	

Tabelle 23 Beschreibung Phase Elaboration 2

Sprint 0

Sprint 0

Dauer	26.03. bis 01.04.2012
Ziele	• Gewonnene Erkenntnisse ins Backlog umsetzen • Backlog Priorisieren
Artefakte	• Product Backlog
Termine	• Sitzung 26.03.2012 • Sitzung 28.03.2012

Tabelle 24 Beschreibung Sprint 0

Sprint 1

Sprint 1

Dauer	02.04. bis 22.04.2012
Ziele	• Prototype für Subsysteme schaffen • Gute Codebasis durch Pairprogramming
Artefakte	• Domainmodell-Prototyp • WCF-Prototyp • Pluginmodell-Prototyp
Termine	• Sitzung 04.04.2012

Tabelle 25 Beschreibung Sprint 1

Sprint 2

Sprint 2

Dauer	23.04. bis 20.05.2012
Ziele	• Featurekomplette Betaversion der Software entwickeln
Artefakte	
Termine	• Sitzung 25.04.2012 • Sitzung 02.05.2012 • Sitzung 09.05.2012

Tabelle 26 Beschreibung Sprint 2

Sprint 3

Sprint 3

Dauer	21.05. bis 08.06.2012
Ziele	• Finale Version der Software Entwickeln • Dokumentation abschliessen
Artefakte	• Dokumentation
Termine	• Sitzung 22.05.2012 • Sitzung 06.06.2012

Tabelle 27 Beschreibung Sprint 3



3.3.5 Sitzungen

Den Betreuern der Studienarbeit soll ein stetiger Einblick in das Projekt gewährleistet werden und dem Team soll die Möglichkeit geboten werden, Fragen zu stellen. Deshalb sollen in regelmässigen Abständen von zwischen einer und zwei Wochen Sitzungen durchgeführt werden.

Der Inhalt dieser Sitzungen muss jeweils in Sitzungsprotokollen festgehalten werden und von allen Parteien zu Beginn der darauffolgenden Sitzung abgenommen werden.

3.3.6 Zeitplanung und -erfassung

Die Abschätzung und Planung erfolgt über die Tickets im Redmine. Über Selbige wird auch die Zeiterfassung realisiert, indem das direkt auf den Tickets gebucht wird.

Redmine bietet umfangreiche Auswertungen, mit denen detaillierte Berichte erstellt werden können.



3.4 Projektorganisation

3.4.1 Team

3.4.1.1 David Schöttl

Kenntnisse
- C, Objective-C, C#, Java
- Unix/OS X, iOS, Windows
- MS Zertifizierungen: .NET 2.0 – 3 x MCTS, 2 x MCPD

Verantwortlichkeiten Architektur & Design, DAL, Webservice, App Config Plugin, Service Manager Plugin

Militärischer Grad Feldweibel (Fw)

Kontakt
David Schöttl
Röslistrasse 3
8304 Wallisellen
079 466 66 14
dave@schoettl.ch

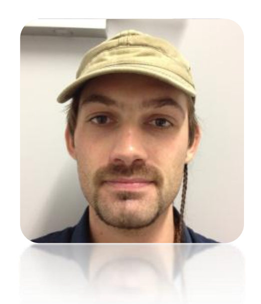


Abbildung 2: David Schöttl

3.4.1.2 Remo Waltenspül

Kenntnisse C#, Java,

Verantwortlichkeiten Plugin Prototyp, Location Plugin, Usability, Personas, Prototypen

Militärischer Grad Soldat (Sdt)

Kontakt
Remo Waltenspül
Moosstr. 2B
8625 Gossau ZH
077 411 03 69
remow88@hotmail.com



Abbildung 3: Remo Waltenspül

3.4.1.3 Diego Steiner

Kenntnisse Ruby, C#, Java, Linux & Bash

Verantwortlichkeiten Scrum-Master, Vorstudie, SQLite Prototyp, Messages-Plugin

Militärischer Grad Obergefreiter (Obgfr)

Kontakt
Diego Steiner
Freudenbergstr. 95
8044 Zürich
079 308 42 73
diego.steiner@u041.ch

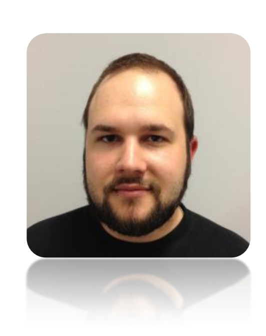


Abbildung 4: Diego Steiner



3.4.2 Betreuung

Thomas Corbat

Thomas Corbat tritt als Hauptbetreuer und Ansprechsperson für Fragen und Probleme auf. Er ist auch im WK vor Ort.

Dr. Markus Stolze

Dr. Markus Stolze tritt als offizieller Experte für die Studienarbeit auf.

3.4.3 Kunde

Von der Seite der Schweizer Armee und der FU Brigade 41 übernimmt der amtierende Kdt Stv des Ristl Bat 20, dessen Stab als "Test User" auftritt, die Aufgaben des Kunden.

Hans-Andrea Veraguth (bisher)

Major Hans-Andrea Veraguth übernimmt in seiner Funktion als Bat Kdt Stv die Rolle des Kunden für das EistCockpit; er verlässt das Ristl Bat 20 im Jahr 2012.

Ramun Berger (neu)

Hauptmann Ramun Berger übernimmt als Nachfolger von Major Veraguth auch seine Rolle als Kunde für EistCockpit.

Michael Klötzli

Obwohl Hauptmann Michael Klötzli nicht mehr direkt zum Ristl Bat 20 gehört, hat er uns auch für Spätere Rückfragen seine Hilfestellung zugesichert. Er tritt als Consultant auf.



3.5 Risikomanagement

Siehe Anhang



3.6 Qualitätssicherung

3.6.1 Review der Tickets

Jedes Ticket wird bei der Erledigung auf den Status „Resolved“ gesetzt. Nach erfolgreichem Review (erfolgt mindestens 1x pro Phase/Sprint) werden die kontrollierten Tickets dann auf den Status „Closed“ gesetzt, was bedeuted, dass das Ticket überprüft wurde. Um einen Sprint abzuschliessen müssen alle Tickets „Closed“ sein.

3.6.2 Review des Codes

Der geschriebene Code selber soll während jedes Sprints durch die anderen Entwickler überprüft und ergänzt werden. In der gesamten Studienarbeit soll auch mindestens einmal ein Codereview mit dem Betreuer stattfinden.

3.6.3 Kriterienliste

Um ein qualitativ Hochwertiges Projekt abzuliefern soll es laufend auf den Massnahmenkatalog in der MoD überprüft werden. Am Ende jedes Sprints wird die MoD gemeinsam durchgegangen und die Umsetzung überprüft.



EistCockpit

4 Vorstudie

David Schöttl, Remo Waltenspül & Diego Steiner



4.1 Dokumenteninformation

Datum	Version	Änderung	Autor
13.03.2012	1.0	Erste Version des Dokuments	rwaltens
27.03.2012	1.1	Interviews hinzugefügt	rwaltens
03.04.2012	1.2	Personas dokumentiert	rwaltens
10.04.2012	1.3	Szenarion dokumentiert	dsteiner
22.04.2012	1.5	Affinities dokumentiert	rwaltens
06.06.2012	1.6	Review	dschöttl



4.2 Allgemein

4.2.1 Zweck des Dokumentes

Ziel dieses Dokumentes ist es die Ist-Situation aufzunehmen und die gegenwärtige Situation zu analysieren. Dabei werden Benutzerbefragungen durchgeführt und aufgrund von den Antworten Personas erstellt.

4.2.2 Gültigkeitsbereich

Dieses Dokument gilt als Grundlage des Projektes und ist daher über die gesamte Projektdauer gültig (20.02 bis 07.06.2012).

4.2.3 Definitionen und Abkürzungen

Siehe „Glossar“.



4.3 Interpretationssession

4.3.1 Interviewvorbereitung

Ziel ist es mit den Interviews, die zukünftige Nutzergruppe zu analysieren und daraus die Anforderungen bezüglich der Features und der Benutzeroberfläche abzuleiten.

4.3.1.1 Identifizierte Interviewpartner

Grundsätzlich existieren zwei Gruppen von Benutzern, die Personen welche in der Triage arbeiten sowie die AdA der Eist Tm.

Für die erste Gruppe Triage wurden folgende drei Personen bestimmt:

- Sdt Meier: Plant Richtstrahlverbindungen sowie Adressierungspläne
- Obwm Moser: Chef der Triage-Gruppe, Koordiniert Übungen
- Wm Müller: Gruppenführer Planungsgruppe

Aus der zweiten Gruppe Eist Tm haben wir den nachstehenden Interviewpartner angefragt:

- Hptm Schmid: Leiter FGG 3 (S3)

4.3.1.2 Termine

- Interview: Di 28.02.2012 21:30 Uhr (Sdt Meier)
- Interview: Di 28.02.2012 22:15 Uhr (Obwm Moser)
- Interview: Di 28.02.2012 22:15 Uhr (Wm Müller)
- Interview: Mi 21.03.2012 21:00 Uhr (Hptm Schmid)



4.3.1.3 Spannende Verhaltensvariablen

- Technisches Knowhow
- Motivation während Arbeit
- Zufriedenheit mit aktuellem Ablauf
- Effizienz bei der Arbeit
- Erfahrungen in diesem Bereich
- Regelmässigkeit der Verwendung
- Genauigkeit/Ausführlichkeit Meldezettel
- Militärisches/Strategisches Denken
- Auffassungsgabe
- Fachliche Kompetenz (Eist Tm)

4.3.1.4 Rollenmatrix

Die nachfolgende Tabelle macht ersichtlich, in welchen Bereichen eine Benutzerbefragung angebracht wäre.

	Triage Mitarbeiter	**Stabsmitarbeiter**
Triage	Beobachtung, sehr wichtig **2x**	Keine Beobachtung, weniger wichtig **0x**
Büro	Keine Beobachtung, weniger wichtig **0x**	Beobachtung, wichtig **2x**

Tabelle 28 Rollenmatrix



4.3.2 Interviews

4.3.2.1 Triage Mitarbeiter 1

Name: *Meier*

Rang: *Soldat*

Funktion: *Richtstrahlplaner*

Was sind deine Aufgaben?

Planen von Übungen, Richtstrahlverbindungen, Adressnetzpläne usw..

Output & Kommunikation

- Telefonanrufe entgegennehmen: Anrufe mit verschiedenen Informationen, Problemen, etc. Telefonate kommen von allen Kompanien.
- Meldezettel ausfüllen: Beschreibung sowie weitere Details zum Anruf werden auf Meldezettel notiert.
- Weiterleiten Meldung: Meldung wird an zuständige Gruppe übergeben: FGG verarbeitet Meldung: Unterschiedliche Meldungen nur zur Info ohne Auftrag, oder mit daraus entstehendem Auftrag.

Benötigter Input?

Meldezettel bestehend aus: Informationen der sechs W-Fragen.

Werdegang/Erfahrung

RS09 Übermittlungsplaner, 3 WK's, Erfahrungen in Triage.

Erfolgsmessung

Wenn keine Meldungen verloren gehen, alles korrekt ausgefüllt wird und Aufträge innert nützlicher Frist abgearbeitet werden.

Probleme bei der Arbeit

Viele Meldungen gleichzeitig, Meldungen gehen unter, Telefonleitungen ausgelastet.

Guter Ablauf

Meldung kommt herein und wird anschliessend effizient abgearbeitet.

Schlechter Ablauf

Telefon wird abgenommen, Meldung unvollständig (ineffizient) oder geht verloren.

Warum ist es schlecht gelaufen?

Zeitdruck, zu viele Aufträge (kein Überblick) und kein Backup.

Wie wird Meldung entgegengenommen, was muss alles gemacht werden?

Meldezettel ausfüllen, Gefechtsjournal Eintrag erstellen.

Was passiert mit diesen Meldungen?

Die Meldung wird der zuständigen Gruppe zugestellt.

Wie viele Meldungen werden pro Stunde empfangen?

5 bis 40.

Wie ist der Umfang einer Meldung(Länge, Dauer eines Gespräches)

10 Sekunden bis 3 Minuten.



Wie ist der Ablauf beim Empfang? Wie effizient läuft so ein Gespräch ab?

Sehr hektisch und wenig Zeit.

Was sind die typischen Inhalte der Meldungen?

Status und Rückrufmeldungen.

Fehlt es dem Meldezettel an Optionen?

Meldezettel verrutscht (schwierig zu beschreiben), mit Maus Werte auswählen, zusätzliche Option an bestimmte Person.

Was ist Ihre Meinung zum bisherigen System?

Gutes System (reicht Kader nicht).

Was für Verbesserungswünsche hätten Sie am System?

Auswahl von Übungen (welche derzeit ablaufen), Kompanien erfassen (mit allen Leuten), Standort von derzeitigen Übungen, Rollen bei System (anschauen, erfassen), Fallback (stündlich ausdrucken).



4.3.2.2 Triage Mitarbeiter 2

Name *Moser*

Rang *Oberwachtmeister*

Funktion *Chef Triage*

Aufgaben

EBIS (Erstellen, Bereithalten, Instandhalten, Sicherstellen), Wachtpläne, Arbeiten für FGG3, Koordination der Planung von Übungen.

Benötigter Input

Meldungen aller Kompanien inkl. Stab.

Werdegang/Erfahrung

Angefangen als IKP, 2 WK IKP als UO, 4 WK Ristl UO, Obwm Lehrgang (Zugführer Stv.).

Erfolgsmessung

Wenn alle einsatzbezogenen Meldungen bei entsprechenden Personen oder Führungsgebiete zeitgerecht angekommen sind.

Probleme bei der Arbeit

Personenmangel (Ressourcenmangel), Überlastung Telefon, zu lange Kommunikationswege (Bsp.: wenn alle in KP-Sitzung sind)

Guter Ablauf

Wenn Ticket kommt wird dies in ein Excel-File geschrieben, Empfänger weiss von Meldung vor dem Erhalt des Meldezettels Bescheid.

Schlechter Ablauf

Person ruft an, Notizen auf "Fresszettel" geht verloren.

Warum ist es schlecht gelaufen?

Zeitdruck.

Arbeitsumgebung

3 Telefone, 2 TEPLAS Client, 1 Fax, abgeschlossener Raum (wegen Telefonen), relativ nahe an FGG3. Siehe dazu „**Error! Reference source not found.**“.

Wie wird Meldung entgegengenommen, was muss alles gemacht werden?

Meldezettel ausfüllen & Eintrag in Gefechtsjournal.

Was passiert mit diesen Meldungen?

Die Meldung wird der zuständigen Gruppe zugestellt.

Wie viele Meldungen werden pro Stunde empfangen?

50 Meldungen/h (in Spitzenzeiten), im Minimum 5 Meldungen /h

Wie ist der Umfang einer Meldung(Länge, Dauer eines Gespräches)

Durchschnittliche Dauer von 30 Sekunden - 15 Minuten, Nachbearbeitung ca. 1 Minute.

Wie ist der Ablauf beim Empfang? Wie effizient läuft so ein Gespräch ab?

Anrufen/Empfang (6 W-Fragen), Emergency Level wird angepasst (Prioritäten).



Was sind die typischen Inhalte der Meldungen?

Status und Rückrufmeldungen.

Fehlt es dem Meldezettel an Optionen?

Nein.

Was ist Ihre Meinung zum bisherigen System?

Meldezettel Punkte schlecht angeordnet.

Was für Verbesserungswünsche hätten Sie am System?

Meldezettel Darstellung ändern Reihenfolge anpassen, Events handeln (TBZ).



4.3.2.3 Triage Mitarbeiter 3

Name: *Meier*

Rang: *Wachtmeister*

Funktion: *Chef Planungsgruppe*

Aufgaben

Planungsgruppe: Kalke, Triage, Schichten Gruppen führen.

Benötigter Input

Technische Feldweibel geben Inputs. Bsp.: Zeichne auf diese Karte das ein.

Werdegang/Erfahrung

Wm in RS als Richtstrahler, Praktikum als MatChef, Wie Ristl Bat 17/2: AGA/an EM08/PANN FGG2; Trotzdem Computervisiert, aber nicht ausgebildet.

Erfolgsmessung

Keine Rückfragen, keine Korrekturen um Auftrag zu erledigen, keine NEF.

Probleme bei der Arbeit

Zum Teil ist es sehr mühsam, da Inputs von vielen verschiedenen Leuten kommen können, 24h Betrieb.

Guter Ablauf

Erhält klaren Auftrag ⇨ Alle nötigen Informationen ⇨ Bearbeiten, Delegieren ⇨ Status melden.

Schlechter Ablauf

Auftrag unklar, mangelnde Infos, Rohmaterial (widersprüchliche Aufträge von verschiedenen Personen.

Warum ist es schlecht gelaufen?

Zeitdruck.

Arbeitsumgebung

Karten, Kalk, Notebook (TEPLAS), Funksystem.

Wie wird ein Auftrag entgegengenommen, was muss alles gemacht werden?

Kalk, Tabellen, Informationen weitergeben (Telefon, Fax, Funk), Eingehende Infos in Tabellen ergänzen, weiterleiten.

Wie viele Meldungen werden pro Stunde empfangen?

Stosszeit: 2 bis 4 Meldungen/Stunde, Minimal 0.3 Meldungen/Stunde (Nacht).

Wie ist der Umfang einer Meldung(Länge, Dauer eines Gespräches)

Dauer ca. 20min, viele kleine, kann stark variieren.

Was für Verbesserungswünsche hätten Sie am System?

klarere Führungsstrukturen (v.a. im KVK), wichtig wäre zum Bsp. Zugführer, Gruppenführer für Gruppen, Flaschenhals für Aufträge.



4.3.2.4 Stabsmitarbeiter 1

Name: *Schmid*

Rang: *Hauptmann*

Funktion: *ehemals Zellenchef FGG3, zur Verfügung Kommandant*

Aufgaben

Verantwortlich Einsatzplanung, Einsatzführung Ristl Bat, Eist Kommandant ("Battle Captain").

Benötigter Input

Stabsarbeitsprozess, APP (Aktionsplanungsprozess), AFP (Aktionsführungsprozess).

Werdegang/Erfahrung

Ristl RS, Ristl Offiziersschule, Ristl Offizier auf System, eingeteilt in Panzerbatallion als Ristl, Chef für Planung (Führung), 5-6 Jahre Kadi in Führungsstaffel Kompanie, anschliessend 3 Jahre als S3 im Einsatz, 2 Jahre Zeitmilitär (1 Jahr Kommandostab, 1 Jahr Fachspezialist UOS, Übermittlungsoffizier bei Infanterie Batallion (insgesamt über 1000 Diensttage).

Erfolgsmessung

Wenn Netz zeitgerecht und stabil dem Endbenutzer bereitgestellt werden kann.

Probleme bei der Arbeit

"Man sieht nur bis zur nächsten Betonmauer", man sieht nicht was draussen läuft, ⇨ Triage ist wichtiger Knotenpunkt (Gateway zur Aussenwelt).

Guter Ablauf

Meldungen die eingehen werden effizient verarbeitet und an zuständige Personen schnell weitergeleitet.

Schlechter Ablauf

"Meldung es geht nicht", unvollständige Meldungen, falsche Informationen, alte TEPLAS Meldungen zu wenig präzis, Grund zu wenig spezifiziert.

Warum ist es schlecht gelaufen?

Zeitdruck, Mitarbeiter zu wenig informiert.

Arbeitsumgebung

In Einsatzstelle mit Blick auf die Karte.

Wie kommt eine Meldung rein?

Über Triage per Mitarbeiter, per Fax.

Wie viele Meldungen kommen pro h rein?

Geballt ⇨ TBZ, viele auf einmal.

Stört dieser Unterbruch den Arbeitsfluss? Wenn ja, was wäre besser?

Nicht massgeblich, gut wäre jedoch eine Priorisierung und Anzeige nur relevanter Meldungen.

Wie reagieren Sie typischerweise auf eine Meldung? Ablauf?

Meldung wird gelesen, eventuell entsteht daraus wiederum Auftrag an Triage.



Gibt es Prioritätsstufen bei diesen Meldungen?

Die momentan eingesetzten Meldezettel besitzen keine Priorisierungsmöglichkeiten.

Was sind die typischen Inhalte der Meldungen?

Status und Rückrufmeldungen, IMFS aufgebaut, Standort erreicht.

Fehlt es dem Meldezettel an Optionen?

Prioritäten (Prioritäten setzen durch den der es bearbeitet).

Was ist ihre Meinung zum bisherigen System?

Funktioniert, könnte aber mit einer Software verbessert werden.

Was für Verbesserungswünsche hätten Sie am System?

- Ablauf à la Call Center mit Instruktionen detailliert beschreiben (Benutzer kennt Ablauf)
- Es werden i.d.R. nicht mehrere Empfänger angegeben (nicht mehrere Meldezettel)
- Anrufer gibt Namen durch, Telefonnr. in System erfasst kann direkt ausgewählt werden
- Empfangene Fax-Dokumente einscannen (Meldungen)
- Delegieren falls Kommandant nicht erreichbar, weiterleiten an Stellvertreter
- Tracking (Bsp.: Alle Meldungen der letzten 30 Minuten anzeigen)
- Priorisierung der Meldungen (in hektischen Situationen), anhand erwartetem eingehenden Uhrzeit
- History (Meldungen bündeln, "Frage bereits einmal angerufen, wegen ähnlichem Problem")
- Feedback (erledigt, nicht erledigt)
- Protokollauszug für höheres Kader ("was wurde erfüllt, was wurde nicht erfüllt")
- Eist 2nd Level Support Erweiterung Fragekatalog durch Fachspezialisten
- Gutes Filtersystem (wichtige von unwichtigen Meldungen unterscheiden)
- Pop-Up Meldungen, die nach gewisser Zeit auftauchen (Erinnerungen)
- Triage sieht auf einen Blick erreichbare Personen.



4.3.3 Arbeitsplatz

4.3.3.1 Triage

Zu der Hauptinfrastruktur in der Triage gehören sicherlich die drei Telefone sowie mindestens ein Laptop. Weiter wird noch ein Fax eingesetzt sowie diverse Informationsplakate von laufenden Übungen sowie Informationen für die Mitarbeiter in der Triage. In der Regel befindet sich die Triage in einem abgeschlossenen Raum, damit die weiteren Mitarbeiter nicht durch die eingehenden Anrufe abgelenkt werden. Zusätzlich stehen natürlich noch normale Büroutensilien wie Stifte, Radiergummis, Klebeband, etc. zur Verfügung.

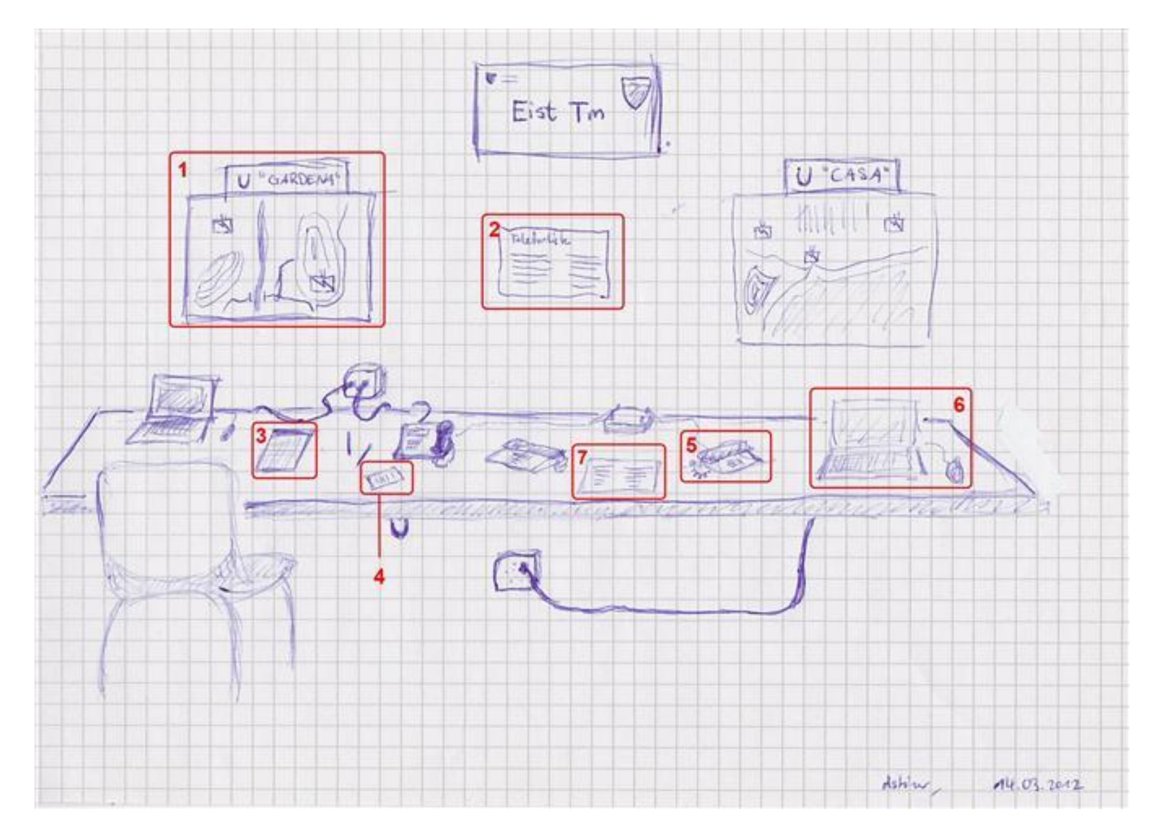


Abbildung 5: Arbeitsplatzskizze

4.3.3.2 Auffälligkeiten

1: Plakate mit Informationen zu aktuellen Übungen

Zu jeder Übung die momentan gerade läuft, existiert ein Plakat auf dem die Standorte der Truppen sowie weiteren Informationen eingetragen sind. Die Triage Mitarbeiter benutzen diese Informationen eher spärlich, aber es kann vorkommen, das man den nächstgelegenen Stao ausfindig machen muss.

2: Telefonliste

Auf der Telefonliste sind die Nummern der wichtigsten Personen in der ZSA aufgelistet. Neben der an der Wand aufgehängten Liste liegt auch noch eine Kopie auf dem Tisch selber, da diese sehr frequent benutzt werden.



3: Notizblock

Der Notizblock dient dem Mitarbeiter dazu zusätzliche Informationen zu notieren. Beispiel dafür könnte ein vom Zellenchef erteilter Telefonauftrag sein oder ein Entwurf eines Meldezettels, der noch ins Reine geschrieben werden muss.

4: Meldezettel:

Beim Meldezettel handelt es sich um das elementare Artefakt bei der Erfassung von Meldungen. Es steht zu jeder Zeit ein Bündel Meldezettel bereit. Siehe „**Error! Reference source not found.**“

5: Telefon

In der Triage bestehen drei Telefonleitungen, somit können zur gleichen Zeit drei Anrufe entgegengenommen werden.
Beim Telefon handelt es sich um ein speziell für militärische Zwecke konzipiertes Gerät, dessen Ziel es ist, allfällige Abhörversuche mittels einem Schalter direkt am Telefonhörer zu verhindern.

6: Laptop

Meistens sind zwei Laptops in der Triage vorhanden, welche hauptsächlich für die Erfassung der Meldungen im Excel benötigt werden (anstatt analog im Gefechtsjournal). Falls jedoch keine neue Anrufe eingehen, wird das Notebook auch häufig als Unterhaltungsmedium genutzt.

7: Kopie Telefonliste

Bei dieser Telefonliste handelt es sich um eine Kopie der Liste, welche an der Wand hängt.

4.3.3.3 Hintergrundinformationen

- Auf dem Tisch liegen in der Regel einzelne Notizzettel mit Informationen von Vorgesetzten. Diese belaufen sich meistens auf circa 2 bis 5 Zettel. Beispielhafte Informationen wären: Hinweis zu einer erwarteten Meldung, bestimmte Person nach einer bestimmten Zeit zurückrufen etc.
- Notizzettel werden nicht mit einer Priorität versehen, dadurch können sie auch nicht nach Wichtigkeit unterschieden werden.
- Für die Notizen gibt es kein bestimmtes Zeitfenster in dem die Zettel abgearbeitet werden müssen. Demnach können Notizen vergessen oder auch verspätet bearbeitet werden.



4.3.4 Affinity

4.3.4.1 Auswertung Affinity Wand

Übersicht

Die folgende Abbildung zeigt die Übersicht über die Affinity Darstellung mittels Post-It Zettel.

Die weiteren Kapitel nehmen Bezug auf die einzelnen Kategorien.

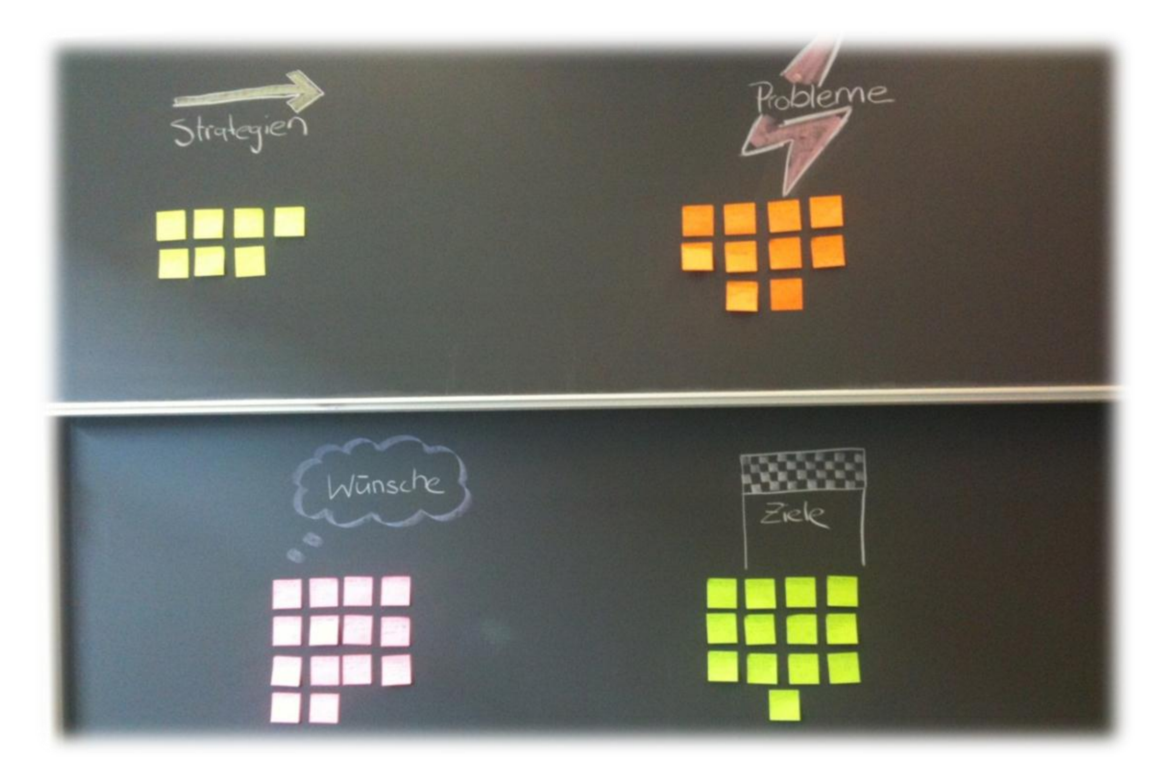


Abbildung 6: Affinity Wand



Probleme

Dieser Abschnitt enthält eine Illustration, mitsamt allen Post-It Zettel, die das Thema "Probleme" betreffen.

Probleme sind Hindernisse, die bei der Erfüllung der Ziele auftreten. Mit der zukünftigen Lösung sollten die Probleme vermieden oder abgeschwächt werden.

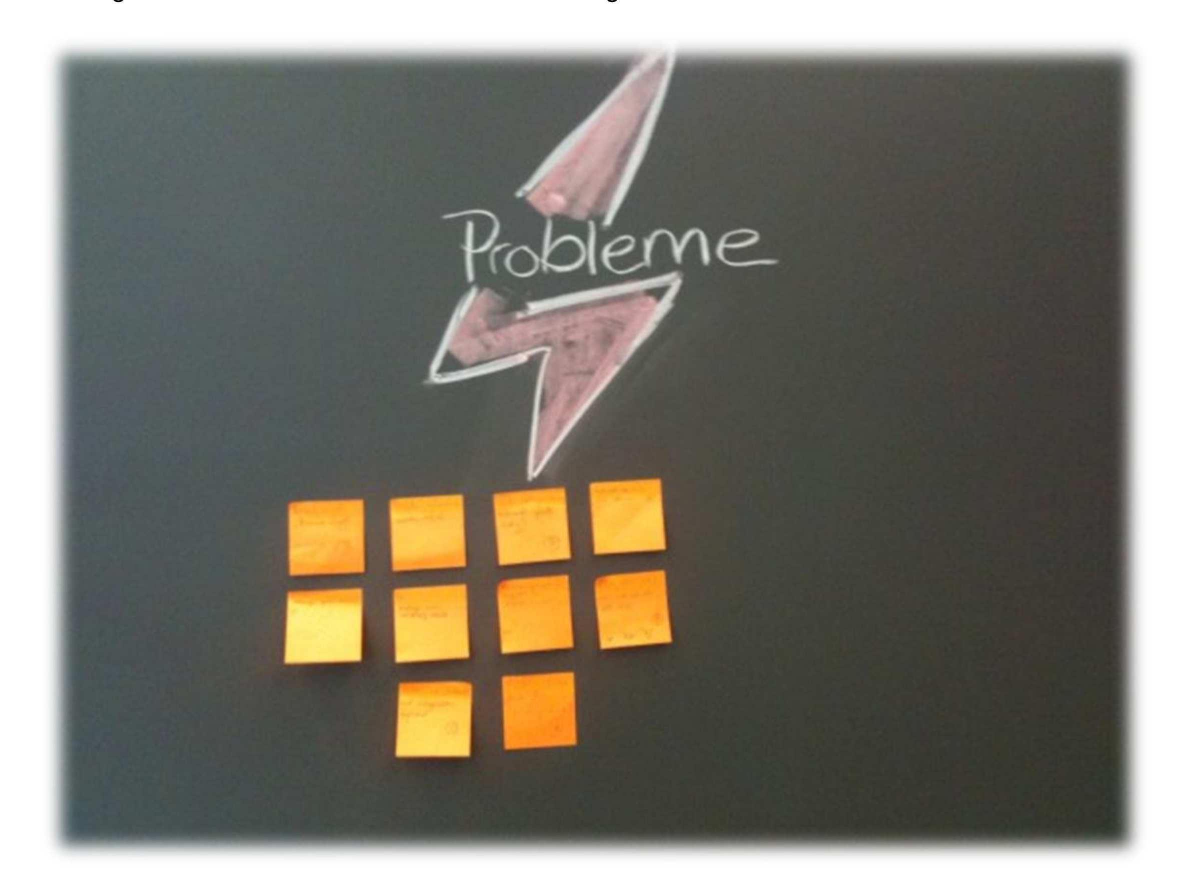


Abbildung 7: Affinity Probleme

Problem	**#**
Mitarbeiter sind zu wenig informiert	4
Man sieht nicht was ausserhalb der ZSA läuft	4
Telefonleitungen sind ausgelastet, hoher Zeitdruck	1
Meldezettel verrutschen beim Schreiben	1
Meldungen werden nicht vollständig erfasst	1
Viele Meldungen kommen miteinander (geballte Ladung)	4
Felder auf Meldezettel sind unangemessen angeordnet	2
Meldungen gehen verloren	1
Personenmangel (Ressourcenmangel)	2
Falsche Informationen werden erfasst	3

Tabelle 29 Auflistung Affinity Probleme



Ziele

Dieser Abschnitt beinhaltet die Ziele, die der Benutzer durch das betätigen des Programms erreichen will.



Abbildung 8: Affinity Ziele



Ziel	#
Effizienteres Arbeiten in der Triage	1
Meldungen können gefiltert und sortiert werden (für Überblick, Kontext, Suche)	3
Positionierung der Anordnung auf dem Meldezettel muss dem Gesprächsverlauf entsprechen	2
Meldungen gehen nicht verloren	1
Meldungen werden immer vollständig erfasst	1
Alle Meldungen kommen beim Empfänger an (Triage ist das Auge der Eist)	3
Alle Details des Meldezettels werden ausgefüllt (W-Fragen)	3
Meldungen können mit Prioritätsindikatoren versehen werden	2
Feedback erledigt, nicht erledigt	3
Meldungen kommen zeitgerecht an	2
Mitarbeiter sind besser informiert, System leitet die Mitarbeiter durch den Prozess	3
Meldungen können an Gruppen und Funktionen übermittelt werden anstatt nur an 1 Person	1
Meldungen Bi-direktional, können weitervermittelt/erweitert werden	3
Status/Kontext eines Staos durch Anzeige deren letzten Meldungen	3
Alle Meldungen werden zentral, digital erfasst	4
Aufträge aus Meldungen generieren	4
Meldungen erreichen immer den gewünschten Adressaten oder dessen Stellvertreter und können abgearbeitet werden.	3

Tabelle 30 Auflistung Affinity Ziele



Strategien

Die Strategien beschreiben bestimmte Abläufe welche durch die Mitarbeiter durchgeführt werden, um ein bestimmtes Ziel zu erreichen. Dadurch wird sichtbar, welches die Kernprozesse im täglichen Betrieb sind.



Abbildung 9: Affinity Strategien



Strategie	#
Meldung wird vom Mitarbeiter erfasst und einer Zelle bzw. einer Funktion in der Zelle weitergeleitet (Anstatt einer Einzelperson) *Ziel: Meldungen kommen zeitgerecht an*	2
Mitarbeiter nehmen Anruf entgegen und müssen in einem ersten Schritt die richtigen Informationen anhand der Plakate ausfindig machen. *Ziel: Meldungen werden vollständig erfasst*	4
Meldezettel ausfüllen, anschliessend Meldung in der Excel-Datei erfassen *Ziel: Meldungen werden vollständig erfasst*	1
Empfänger liest Meldung, interpretiert sie und erstellt neuen Auftrag *Ziel: Auftrag aus Meldung generieren*	4
Abteilungsleiter stellen Informationen zusammen, die für die Mitarbeiter relevant sind, anschliessend werden Mitarbeiter informiert *Ziel: Mitarbeiter besser informiert*	3
Damit Meldungen ankommen die wichtig sind, wird Meldung direkt mündlich überbracht *Ziel: Meldungen gehen nicht verloren*	1
Alle sechs W-Fragen müssen ausgefüllt werden, es gibt keine Ausnahmen *Ziel: Meldungen vollständig*	2
Benutzer wählt anhand von Kriterien aus welche Meldungen er anzeigen möchte *Ziel: Meldungen können gefiltert werden*	3
Meldung auswählen, damit verbundene Nachfolgemeldungen (Gruppierung) anzeigen *Ziel: Status/Kontext eines Stao durch Anzeige der letzten Meldungen*	3

Tabelle 31 Auflistung Affinity Strategien



Wünsche

Bei den Wünschen handelt es sich um eine Auflistung von Anregungen für die zu entwickelnde Anwendung der befragten Mitarbeiter. Aus den Wünschen können zum Teil Ziele entstehen, die von der Anwendung erfüllt werden müssen.



Abbildung 10: Affinity Wünsche



Wunsch	#
Priorisierung von Meldungen	2
Filtern von Meldungen	4
Telefon entgegennahme durch dynamisches Prozessschema bestimmt. Anrufer wird step-by-step abgefragt.	3
Wunsch einsetzten um Prozesse zu optimieren	4
Auto Completion beim Erfassen der melderelevanten Daten	4
Zusätzliches Empfängerfeld soll möglich sein	2
Feedback (Meldung erledigt/nicht erledigt)	4
Mehrere Meldeempfänger müssen möglich sein	1
History der letzten Meldungen	3
Triage Mitarbeiter sollen weniger ausgelastet sein	1
Meldungen müssen zeitgerecht beim Empfänger ankommen	2
Empfangene Faxmeldungen einscannen	4
Erreichbarkeit der Personen in der ZSA sichtbar	2
Meldung an eine bestimmte Funktion in der Zelle und nicht an Person gebunden	2

Tabelle 32 Auflistung Affinity Wünsche



4.3.5 Root Cause Analyse

4.3.5.1 Probleme

Schadensklasse	Beschreibung
hoch	Hat grosse Auswirkungen, stört die täglichen Abläufe erheblich, ev. mit finanziellen Folgen verbunden
mittel	Betrifft kleineren Bereich, keine grösseren finanziellen Verluste
niedrig	Keine finanzielle Verluste, betrifft nur kleinere Gruppe, kleines Schadenspotenzial

Tabelle 33 Root Cause Problem Schadensklassen

4.3.5.2 Auflistung

Die Eintrittswahrscheinlichkeit von den geschilderten Problemen gehen aus den durchgeführten Interviews sowie aus Schätzungen hervor.

#	Problem	Standort	Zeitpunkt	Schadens-ausmass	Eintritts-wahrschein-lichkeit
1	Meldung unvollständig	Triage	Während normalem Tagesablauf	niedrig	20%
2	Meldung geht verloren	Triage	Während normalem Tagesablauf	mittel	5%
3	Unklar wer Meldungsempfänger	Triage	Während normalem Tagesablauf	niedrig	10%
4	Übertragungszeit Meldungen zu lange	Triage	Während normalem Tagesablauf	mittel	15%
5	Meldungen enthalten fehlerhafte Informationen	Triage	Während normalem Tagesablauf	mittel	5%
6	Meldungen bleiben liegen	Triage	Während normalem Tagesablauf	niedrig	10%

Tabelle 34 Auflistung von Problemen



4.3.5.3 Ursachen-Wirkung

Ursache	Wirkung
Hektische Situation, fehlendes Nachfragen, Mitarbeiter zu wenig informiert	Meldung unvollständig
Mitarbeiter arbeiten nachlässig und sind unachtsam	Meldung geht verloren
Fehlende Mitarbeiterinformation (Einführung), keine oder fehlende Unterlagen	Unklar wer Meldungsempfänger
Weiterleitung an falsche Person, Mitarbeiter schlecht informiert, Kompetenzen nicht klar geregelt	Übertragungszeit von Meldungen zu lange
Unsorgfältige Arbeitsweise von Mitarbeitern, bei Unklarheiten wird nicht nachgefragt	Meldungen enthalten fehlerhafte Informationen
Meldungen werden an Personen statt Positionen weitergereicht, Erreichbarkeit von Personen in ZSA nicht bekannt	Meldungen bleiben liegen

Tabelle 35 Ursachen-Wirkungstabelle



4.3.5.4 Lösungen zur Problemvermeidung

Problem 1: Meldung unvollständig

- Es wird programmatisch verifiziert, ob sämtliche notwendige Daten eingegeben wurden.
- Pflichtfelder, welche zwingend ausgefüllt werden müssen

Problem 2: Meldung geht verloren

- Meldungen, welche noch nicht vollständig erfasst worden sind, können zwischengespeichert werden.
- Da Meldungen elektronisch erfasst werden, ist es unmöglich, dass diese verloren gehen.

Problem 3: Unklar wer Meldungsempfänger

- Hilfestellung auf Formular zum Erfassen der Meldungen
- Weiterleiten an verantwortlichen Zellenchef (sollte klar sein, da Zuständigkeitsbereiche von Zellen bekannt sind)
- Mitarbeiter vor Antritt der Arbeit gut informieren

Problem 4: Übertragungszeit Meldungen zu lange

- Durch die elektronische Lösung wird die Übertragungszeit auf ein Minimum reduziert.

Problem 5: Meldungen enthalten fehlerhafte Informationen

- Explizit in Formular erwähnen, dass bei Unsicherheiten besser nochmals nachgefragt werden soll.

Problem 6: Meldungen bleiben liegen

- Zu jeder Person sollten Stellvertreter registriert werden, an die die Nachricht gesendet wird, falls primärer Empfänger nicht erreichbar ist.
- Wenn Meldung lange ungelesen in der Übersicht liegt, soll sie hervorgehoben werden.



4.4 Behaviour Pattern

Für die Erstellung der Persona Linien und Interview Punkte haben wir sämtliche Interviews ausgewertet. Anhand der gewonnen Erkenntnisse wurden danach essentielle Verhaltensvariablen definiert. Die befragten Personen wurden in einem weiteren Schritt nach ihren Fähigkeiten auf einer Skala eingetragen.

4.4.1 Interview Punkte

Bezeichnung	Person
A	Triage Mitarbeiter (Sdt Müller)
B	Triage Gruppenführer (Obwm Moser)
C	Gruppenführer Planungsgruppe (Wm Meier)
D	Eist TM Mitarbeiter (Hptm Schmid)

Tabelle 36 Interviewpartner

Technisches Knowhow

C D B A
klein gross
Technisches Know-How

Abbildung 11 Technisches Know-How

Motivation während Arbeit

A B C D
klein gross
Motivation während Arbeit

Abbildung 12 Motivation während der Arbeit

Zufriedenheit mit aktuellem Ablauf

A B D C
klein gross
Zufriedenheit mit aktuellem Ablauf

Abbildung 13 Zufriedenheit mit aktuellem Ablauf

Effizienz bei der Arbeit

C D A B
klein gross
Effizienz bei der Arbeit

Abbildung 14 Effizienz bei der Arbeit



Erfahrungen in diesem Bereich

C A D B
klein — gross
Erfahrungen in diesem Bereich

Abbildung 15 Erfahrungen in diesem Bereich

Regelmässigkeit der Verwendung

C D B A
selten — häufig
Regelmässigkeit der Verwendung

Abbildung 16 Regelmässigkeit der Verwendung

Genauigkeit/Ausführlichkeit Meldezettel

C A B D
oberflächlich — detailliert
Genauigkeit/Ausführlichkeit Meldezettel

Abbildung 17 Genauigkeit Meldezettel

Militärisches/Strategisches Denken

A B C D
schwach — stark
Militärisches/Strategisches Denken

Abbildung 18 Militärisches/ Strategisches Denken

Auffassungsgabe

A C B D
schlecht — gut
Auffassungsgabe

Abbildung 19 Auffassungsgabe

Fachliche Kompetenz (Eist Tm)

D C A B
schwach — stark
Fachliche Kompetenz (Eist Tm)

Abbildung 20 Fachliche Kompetenz

Konzentrationsfähigkeit

A B C D
schwach — stark
Konzentrationsfähigkeit

Abbildung 21 Konzentrationsfähigkeit



4.4.2 Persona Linien

Technisches Know-How

Motivation während Arbeit

Zufriedenheit mit aktuellem Ablauf

Effizienz bei der Arbeit

Erfahrungen in diesem Bereich

Regelmässigkeit der Verwendung

Genauigkeit/Ausführlichkeit

Militärisches/Strategisches Denken

Auffassungsgabe

Fachliche Kompetenz (Eist Tm)

Konzentrationsfähigkeit

Abbildung 22 Persona Linien

4.4.2.1 Fazit

Die drei interviewten Benutzer, welche in der Triage arbeiten, haben wir zu zwei Personas zusammengefasst. Zum einen wäre das die Gruppe der Soldaten, zum anderen die Gruppe der Führungskräfte. Aus diesem Grund resultieren drei Personas für unser System.



4.5 Personas

4.5.1 Persona Frank Fleissig

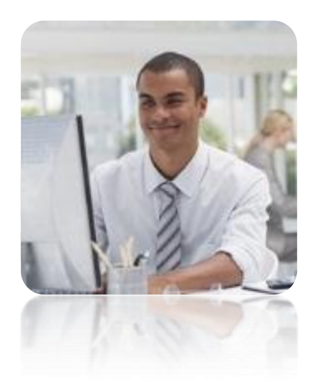


Abbildung 23: Persona Frank Fleissig

Alter	*24 Jahre*
Zivilstand	*Ledig, keine Kinder*
Ausbildung	*Kaufmännische Lehre*
Tätigkeit	*Sachbearbeiter in KMU*

Technische Kenntnisse Routinierter Umgang mit dem Computer, oberflächliche Technikkenntnisse

Funktion Frank ist ein Soldat und leistet seinen Dienst einmal pro Jahr, i.d.R. ist er deswegen drei (bzw. vier Wochen bei einem KVK Einsatz) im Einsatz. Er ist zuständig für die Entgegennahme und Weiterleitung von Meldungen per Telefon.

Arbeitskontext Der Arbeitsort von Frank ist in der Regel eine ZSA oder eine geheime Bunkeranlage, wichtig ist zudem, aufgrund von vielen Telefonaten ein abgeschlossener Raum nahe der FGG 3. Das Arbeitszimmer ist mit büroüblichen Mitteln(drei Telefone, zwei TEPLAS Client,ein Fax, Büroutensilien etc.) eingerichtet. Der Lärm sollte eigentlich kein Problem darstellen, da die Triage-Abteilung in einem abgeschlossenen Raum sich befindet. Unterbrüche können im Falle eines Schichtwechsel oder Fragen von verschiedenen Personen entstehen.

Arbeitsstil Frank verrichtet seine Arbeit in der Regel pflichtbewusst und zuverlässig. Er ist zudem ziemlich erfahren, da er normalerweise schon mehrere Wiederholungskurse absolviert hat. Es kann jedoch auch vorkommen, dass er einzelne Meldungen aufgrund von hektischen Situationen zu wenig detailliert erfasst. Dies stellt jedoch nicht den Regelfall dar, normalerweise nimmt er die Daten genau auf und leitet sie per Meldezettel weiter. Auf der Tastatur bringt er am Telefon eine durchschnittliche Anschlagsrate von 120 ApM hin.

Persönlichkeit & Vorlieben Die Abläufe in der Triage sind Frank geläufig, er kann sich zudem gut in ein Team integrieren, da er eher der gesellige, kommunikative Typ ist. Aufgrund seiner offenen Art neigt er auch dazu sich schnell durch andere Personen ablenken zu lassen. Für ihn ist primär wichtig, dass das soziale Umfeld und die Chemie zwischen den Mitarbeitern stimmt. Weiter hätte er gerne eine bessere Übersicht über den Status der verschiedenen Meldungen sowie eine bessere Übersicht.



Vorkenntnisse & Lernen	Frank hat schon mehrere Wiederholungskurse absolviert und ist deshalb ziemlich routiniert und vertraut mit den Abläufen in der Triage. Seine Computerkenntnisse umfassen nur die Grundlagen, jedoch ist dies auch keine primäre Anforderung um in der Triage zu arbeiten, da es mehr von Vorteil ist, wenn man kommunikativ stark ist. Zu seinen Stärken gehören die schnelle Auffassungsgabe sowie die Flexibilität neue Sachverhalte zu erlernen.
Pain Points & Frustrationen	Es gibt verschiedene Punkte die Frank am derzeitigen System stören. Zum einen wäre ein ganz banales Problem beim Beschreiben der Meldezettel am Telefon. Sobald er ein Telefon entgegennimmt, hat er nur noch eine Hand frei um die Informationen zu notieren, dadurch rutscht der kleine Meldezettel andauernd weg. Ein weiterer Punkt ist die Anordnung der Optionen auf dem Meldezettel, die relevantesten Infos stehen nicht an oberster Stelle. Weiter kann es in hektischen Situationen dazu kommen, dass einzelne Meldungen verloren gehen. Dies sollte eigentlich nie passieren und ist inakzeptabel, trotzdem kann es schnell vorkommen, aufgrund der kleinen Grösse der Zettel.
Eigenschaften & Behaviour Variables	Technisches Knowhow, Motivation während Arbeit, Zufriedenheit mit aktuellem Ablauf, Effizienz bei der Arbeit, Erfahrungen in diesem Bereich, Regelmässigkeit der Verwendung, Genauigkeit/Ausführlichkeit Meldezettel, Militärisches/Strategisches Denken, Auffassungsgabe, Fachliche Kompetenz (Eist Tm)
Ziele	Effizienz der Meldungen verbessern, Verlieren von Meldezetteln vermeiden, Abläufe optimieren, Alle notwendigen Informationen auf Zettel



4.5.1.1 Day-in-the-Live Szenario

Der Tag beginnt für Frank um ca. halb sieben Uhr. Er geht duschen, zieht sich an und geht anschliessend um ca. sieben Uhr Morgenessen. Nach dem Morgenessen fängt er mit seiner üblichen Arbeit um ca. halb acht Uhr an. Er spricht kurz mit seinen Kameraden, und tauscht im schnell Durchgang noch die wichtigsten Infos aus, bevor es richtig losgeht. Kurz vor der Übernahme der Arbeitsschicht informiert der Wachtmeister Iwo Informiert noch alle Soldaten kurz in einem Rapport und gibt ein paar gute Inputs und Tipps für den Arbeitstag. Schon geht es los, die Meldezettel sind vorbereitet und die ersten Telefone klingeln. Frank nimmt den Hörer ab, beim Anrufer handelt es sich um einen Soldaten der mit einer Gruppe von Richtstrahlpionieren; eine Richtstrahlantenne aufgebaut hatte und soeben den Standort erreicht hat. Er nimmt alle Daten entgegen und versucht sie zeitgleich auf dem Meldezettel zu notieren. Dies gestaltet sich als Herausforderung, da der Zettel andauernd verrutscht und keine Hand übrig bleibt um den Meldezettel zu fixieren, da die zweite Hand den Knopf am Hörer gedrückt halten muss (siehe „“). Der Soldat beendet das Telefonat, dies ermöglicht Frank die restlichen Informationen, welche er noch nicht erfasst hat zu notieren. Gerade als er die letzten Wörter aufgeschrieben hat, kommt ein weiterer Anruf herein. Leider bleibt nicht genügend Zeit, um den Zettel vom vorherigen Telefonat an die vorgesehene Abteilung weiterzuleiten. Erneut nimmt er die Daten vom derzeitigen Gespräch auf und versucht schon während dem Anruf möglichst viel zu notieren. Den Namen hat er nicht ganz verstanden, deshalb fragt er noch einmal nach, der anrufende Wachtmeister buchstabiert seinen Namen, welcher Frank sogleich auf den Zettel schreibt. Die Zeit vergeht wie im Flug, da an diesem Morgen viele Anrufe reingekommen sind und ein hektisches Treiben geherrscht hat. Plötzlich erinnert sich Frank an den ersten Meldezettel, den er aus lauter Arbeitseifer vergessen hat weiterzureichen. Umgehend nimmt er den übersehen Meldezettel und bringt ihn sogleich der zuständigen Person. Der Empfänger - ein Hauptmann der FGG3, rügt Frank für die verschlampte Meldung, da er auf eine schnelle Meldung angewiesen war. Er ermahnt ihn, die Arbeit gefälligst seriös zu nehmen und Meldungen umgehend weiterzuleiten. Nach diesem Malheur ist es auch schon Zeit für das Mittagessen, welches Frank mit seinen Kameraden einzunehmen Pflegt. Zwecks des 24h Betriebes bleibt jeweils mindesten 1 Mitarbeiter in der Triage, bis ein Anderer zurückkommt um ihn abzulösen. Für Informationsaustausch besteht im Normalfall kein Bedarf, da die Aktivitäten jeweils zuerst fertig abgearbeitet werden und es sich immer um kleinere Pakete handelt. Direkt nach dem Mittagessen ist meistens nicht so viel los, was auch an diesem Tag nicht anders ist. Deshalb lenkt er sich ein bisschen ab mit der aktuellen Tageszeitung. Just in diesem Moment beginnen die Telefone wieder zu klingeln. Frank geht deshalb wieder seinen normalen Tätigkeiten nach und nimmt Telefone entgegen und schreibt sich die Meldungen auf die Meldezettel auf.



Plötzlich kommt ein höherer Offizier und fragt, ob eine bestimmte Meldung einer Richtstrahlgruppe bereits eingegangen ist, und falls nicht ob man ihn benachrichtigen könne, sobald sie aufgenommen wird. Ein Soldat nickt, notiert es auf einen Notizzettel und legt ihn auf den Tisch. Circa zwei Stunden später kommt wie erwartet die erhoffte Meldung. Frank meldet dies dem Offizier, welcher sich für die schnelle Information bedankt.

Am späten Nachmittag kommen noch ein paar weitere Anrufe herein, die alle ordnungsgemäss abgearbeitet werden. Kurz vor dem Abendessen erhält Frank noch den Auftrag ein Dokument an eine andere Kompanie zu faxen. Nach dem Fax kommt ein Kamerad, der ihn aufmerksam macht, dass es Zeit für das Abendessen ist. Zusammen rekapitulieren sie nochmals kurz den Arbeitstag während der Einnahme des Abendessens und unterhalten sich locker. Nach dieser kurzen Pause macht er sich noch an die Tagesschlussetappe bis zum Schichtwechsel um 22 Uhr. Während dieser Zeit sind keine aussergewöhnlichen Meldungen mehr hereingekommen. Gegen zehn Uhr am Abend kommt wie geplant die Ablösung, welche die Nachtschicht übernehmen wird. Ab diesem Zeitpunkt hat Frank den verdienten Feierabend erreicht. Er macht sich auch schon bald bereit um schlafen zu gehen.



4.5.1.2 Ist-Szenarien

Ist-Szenario 1 Nachricht entgegennehmen:

Ausgangssituation	Frank ist in der Triage am Arbeiten und wartet auf den nächsten Anruf. Seinen Meldezettel hat er bereitgelegt, dass er bei einem Anruf direkt Notizen machen kann.
Auslöser	Anruf eines Stao-Chefs an die Triage
Schritte	Frank nimmt das Telefon entgegen und schreibt alle benötigten Informationen auf den Meldezettel. Sobald er das Telefonat beendet hat, füllt er die restlichen Felder des Zettels aus. Anschliessend leitet er den Meldezettel weiter an die zuständige Person, indem er den Zettel in das passende Fach (bzw. Briefkasten) einwirft.
Probleme	Meldezettel ist nicht vollständig ausgefüllt, Meldezettel gehen verloren (Verweis Interview 1 Probleme bei der Arbeit)
Ziele	Die Meldung wird schnell aufgenommen, verarbeitet und weitergeleitet.

Ist-Szenario 2 Fehlende Informationen auf Meldezettel:

Ausgangssituation	Frank ist in der Triage am Arbeiten und wartet auf den nächsten Anruf. Seinen Meldezettel hat er bereitgelegt, so dass er bei einem Anruf direkt Notizen machen kann.
Auslöser	Anruf eines Stao-Chefs an die Triage, Hektische Situation, wenig Schlaf
Schritte	Frank nimmt das Telefon entgegen und schreibt alle benötigten Informationen auf den Meldezettel. Der Meldezettel wird nicht vollständig ausgefüllt, wichtige Felder fehlen. Anschliessend leitet er den Meldezettel weiter an die zuständige Person, indem er den Zettel in das passende Fach (bzw. Briefkasten) einwirft. Der Empfänger sieht, dass essentielle Infos nicht vorhanden sind und retourniert die Meldung zurück an die Triage. Falls keine Kontaktdaten angegeben wurden, müssen die vergessenen Daten mühsam rekonstruiert werden.
Probleme	Meldezettel ist nicht vollständig ausgefüllt, Effizienz der Meldungen schlecht, Lange Dienstwege (Verweis Interview 1 Schlechter Ablauf)
Ziele	Die Meldung soll sämtliche benötigten Informationen enthalten.

Ist-Szenario 3 Aus Meldung wird Auftrag generiert

Ausgangssituation	Frank hat eine Meldung einer Person zugestellt.
Auslöser	Die auf dem Meldezettel stehende Information impliziert einen neuen Auftrag
Schritte	Der Empfänger liest die Meldung. Daraus entsteht ein Auftrag



wiederum an die Triage, einer bestimmten Kompanie anzurufen und etwas mitzuteilen.

Probleme Die Distanz zwischen FGG3 und Triage führt zu einer Verspätung der Meldungen.

Ziele Der Meldungsfluss zwischen den verschiedenen Abteilungen und der Triage soll beschleunigt werden.

Ist-Szenario 4 Nachfrage Name

Ausgangssituation Frank hat soeben ein Telefon entgegengenommen, wobei der Anrufer ihm bereits den Namen geschildert hat.

Auslöser Anruf eines Stao-Chefs an die Triage, Anrufer spricht undeutlich

Schritte Leider hat Frank den Namen nicht ganz verstanden, deshalb weiss er nicht welchen Namen er notieren soll. Aus diesem Grund fragt er nochmals in höflichem Ton nach dem Nachname.
"Können Sie bitte ihren Namen nochmals wiederholen, ich habe ihn leider nicht ganz verstanden - vielen Dank!"
Der Anrufer hat es eilig, darum wiederholt er den Namen nur kurz. Dies reicht Frank jedoch nicht, da der anrufende Soldat zu wenig deutlich gesprochen hat. Er fordert ihn nochmals auf, seinen Familienname langsam zu buchstabieren.

Probleme Vermehrt kommt es zu Verständnisproblemen, weil eine Partei den anderen falsch versteht. (Aus diesem Grund sollten wichtige Informationen bestätigt werden.)

Ziele Meldungen korrekt übertragen, Name auf Meldezetteln richtig schreiben

Ist-Szenario 5 Layout Meldezettel

Ausgangssituation Frank ist in der Triage am Arbeiten und wartet auf den nächsten Anruf. Seinen Meldezettel hat er bereitgelegt, so dass er bei einem Anruf direkt Notizen machen kann.

Auslöser Anruf eines Stao-Chefs an die Triage



Schritte Frank liest die aktuelle Tageszeitung bis plötzlich ein Anruf eingeht, umgehend nimmt er das Telefon entgegen. Er meldet sich mit dem Standardbegrüssungsschema Ristl Bat 20 Soldat Fleissig "Guete Morge", darauf stellt sich auch der Anrufer kurz vor. Anscheinend hat er es ziemlich eilig, da er direkt erwähnt wo er sich derzeit aufhält und um was das es bei der Meldung geht. Frank notiert sich diese Informationen, wobei er sich ein bisschen über die Anordnung der einzelnen Punkte auf dem Meldezettel nervt. Das Datum sowie die Uhrzeit stehen an erster Stelle, jedoch könnte man diese Informationen auch erst nach der Beendigung des Gesprächs ausfüllen. (Verweis Interview 2 Was für Verbesserungswünsche hätten Sie am System?)

Probleme Anordnung der Fragen auf dem Meldezettel verwirrend, Meldezettel bietet nur wenige Informationen

Ziele Meldung vollständig erfassen

Ist-Szenario 6 Falsche Informationen werden erfasst

Ausgangssituation Frank hat soeben einen Anruf entgegengenommen. Ausserdem liegt der Meldezettel für das gegenwärtige Telefonat bereit und die Excel-Liste auf dem Laptop ist geöffnet.

Auslöser Anruf eines Stao-Chefs an die Triage, Hektische Situation in der Triage

Schritte Frank begrüsst den Anrufer mit seiner üblichen Begrüssungsklausel, daraufhin stellt sich auch der Gesprächspartner kurz vor. Leider hat Frank den Namen nicht ganz verstanden, um sicherzustellen, dass er den korrekten Namen auf dem Meldezettel erfasst, fragt er nochmals nach dem Namen. Der anrufende Soldat berichtet ihm daraufhin kurz, dass sie soeben den Standort 2 erreicht haben. Frank bedankt sich für das Telefon und wünscht ihm einen schönen Tag. Nach dem Anruf füllt er noch die verbleibenden Felder aus (Datum, Uhrzeit). Daraufhin überbringt er dem zuständigen Zellenchef die Meldung. Nach einer Weile tritt der der Empfänger der letzten Meldung in die Triage. Er macht Frank darauf aufmerksam, dass die Information "Bereitschaftsraum verlassen" für diese Gruppe nicht zutreffen kann. Frank beteuert seinen Fehler, es handelt sich natürlich um die Meldung "Standort erreicht".

Probleme Schwierig sicherzustellen, dass Informationen korrekt sind

Ziele Meldung ohne Falschinformationen erfassen



4.5.1.3 Soll-Szenarien

Soll-Szenario 1 Meldung erfassen

Ausgangssituation Frank hat soeben seinen Arbeitstag begonnen, und seine Meldezettel bereitgelegt.

Auslöser Anruf eines Stao-Chefs an die Triage

Schritte Es geht ein Anruf ein, der Frank umgehend entgegennimmt. Parallel läuft das Programm EistCockpit, mit welchem er direkt den Namen des Anrufers erfasst. Die Applikation bietet ihm dazu alle im Batallion registrierten AdA zur Auswahl an. Während dem Gespräch kann er zudem direkt den Standort wählen, es werden ihm dazu die gegenwärtigen erfassten Standorte vorgeschlagen. Anschliessend muss er nur noch den Typ der Meldung definieren. Nach dem Aufhängen des Telefons kann Frank noch eine kurze Beschreibung angeben und die Empfängerabteilung festlegen. Das Datum wird automatisch beim Erstellen einer neuen Meldung bestimmt. Sobald alle benötigten Felder ausgefüllt sind, kann der Triage-Benutzer die Meldung versenden.

Probleme Benutzer hat Mühe, während dem Telefonieren das Notebook zu bedienen.

Ziele Meldungen werden immer vollständig ausgefüllt, Die Meldung muss nicht mehr physisch (zu Fuss) verteilt werden



Soll-Szenario 2 Standort erreicht

Ausgangssituation Es ist ein Anruf eingegangen, und der Name wurde bereits aufgenommen.

Auslöser Anruf eines Stao-Chefs an die Triage

Schritte Der Anrufer, ein gewisser Soldat Hülsensack, verlangt den für die FGG3 zuständigen Offizier. Weiter gibt er seinen aktuellen Standort an, und bittet Frank dem zuständigen Offizier eine Meldung auszuhändigen, ihn sobald wie möglich über die folgende IMFS-Telefonnummer zu kontaktieren: 0110-0382. Frank geht Punkt für Punkt durch das Meldeformular der EistCockpit Anwendung, dazu trägt er den Standort, den zuständigen Offizier, sowie den Meldungstyp "Rückruf" ein. Als er das Telefon beendet hat, versucht er die Meldung zu speichern, was aber vom System unterbunden wird, da die Rückruf-Nummer nicht angegeben wurde. Deshalb schreibt er noch kurz die Natelnummer in das vorgesehene Feld und speichert die Meldung.

Probleme Der Benutzer vergisst einzelne Felder auszufüllen

Ziele Meldungen werden immer vollständig ausgefüllt, Typ der Meldung wird klar definiert.

Soll-Szenario 3 Rückruf erwünscht

Ausgangssituation Die Software ist gestartet und Frank bereit für die Aufnahme einer Meldung

Auslöser Anruf eines Stao-Chefs an die Triage

Schritte Es geht ein Anruf ein, der Frank umgehend entgegennimmt. Parallel läuft das Programm EistCockpit, mit welchem er direkt den Namen des Anrufers erfasst. Die Applikation bietet ihm dazu alle im Batallion registrierten AdA zur Auswahl an. Während dem Gespräch kann er zudem direkt den Standort wählen, es werden ihm dazu die gegenwärtigen erfassten Standorte vorgeschlagen. Anschliessend muss er nur noch den Typ der Meldung definieren. Nach dem Aufhängen des Telefons kann Frank noch eine kurze Beschreibung angeben und die Empfängerabteilung festlegen. Das Datum wird automatisch beim Erstellen einer neuen Meldung bestimmt. Sobald alle benötigten Felder ausgefüllt sind, kann der Triage-Benutzer die Meldung versenden.

Probleme Benutzer hat Mühe, während dem Telefonieren das Notebook zu bedienen.

Ziele Meldungen werden immer vollständig ausgefüllt, Die Meldung muss nicht mehr physisch (zu Fuss) verteilt werden



4.5.2 Persona Iwo Informiert



Abbildung 24: Persona Iwo Informiert

Alter	*26 Jahre*
Zivilstand	*Ledig, keine Kinder*
Ausbildung	*Kaufmännische Lehre, Bachelor für Betriebswirtschaft, Lehrlingbetreuer*
Tätigkeit	*Arbeitet in einer Grossfirma als Betriebsökonom*

Technische Kenntnisse Erfahren im Umgang mit Computer, Grundkenntnisse in Informatik

Funktion Iwo hat den Rang Wachtmeister, deshalb ist er auch zuständig für die Führung der Triage-Gruppe. Er koordiniert und leitet Befehle bzw. Aufträge vom höheren Kader weiter. Zudem gibt er dem Team (i.d.R. bestehend aus Soldaten) Inputs und Verbesserungsvorschläge um die Effizienz zu steigern.

Arbeitskontext Der Arbeitsort von Frank ist in der Regel eine ZSA oder eine geheime Bunkeranlage, wichtig ist zudem, aufgrund von vielen Telefonaten ein abgeschlossener Raum nahe der FGG 3. Das Arbeitszimmer ist mit büroüblichen Mitteln(drei Telefone, zwei TEPLAS Client, ein Fax, Büroutensilien etc.) eingerichtet.
Der Lärm sollte eigentlich kein Problem darstellen, da die Triage-Abteilung in einem abgeschlossenen Raum sich befindet. Unterbrüche können im Falle eines Schichtwechsel oder Fragen von verschiedenen Personen entstehen.

Arbeitsstil Iwo bereitet sich seriös auf seine Arbeit vor und übernimmt als Gruppenführer eine Vorbildfunktion. Ihm ist es ein weiteres Anliegen, dass der "Team-Spirit" bzw. Teamgeist gefördert werden kann, was natürlich indirekt wieder zu erhöhten Arbeitsleistungen führen kann. Um dies zu erreichen führt er von Zeit zu Zeit Gespräche mit verschiedenen Soldaten durch, daraus kann er frühzeitig erkennen wo Probleme auftreten. Als Leiter der Triage-Gruppe muss er vor allem die Übersicht bewahren und bei Fragen der Soldaten Auskunft geben können. Dies stellt für ihn jedoch kein Problem dar, da er gut informiert ist.



Persönlichkeit & Vorlieben	Die Abläufe in der Triage sind Frank geläufig, er kann sich zudem gut in ein Team integrieren, da er eher der gesellige kommunikative Typ ist. Aufgrund seiner offenen Art neigt er auch dazu sich schnell durch andere Personen ablenken zu lassen. Für ihn ist primär wichtig, dass das soziale Umfeld und die Chemie zwischen den Mitarbeitern stimmt. Weiter hätte er gerne eine bessere Übersicht über den Status der verschiedenen Meldungen sowie eine bessere Übersicht.
Vorkenntnisse & Lernen	Iwo besitzt Anwenderkenntnisse im Umgang mit dem Computer, tieferes technisches Wissen kann er nicht aufweisen. In der Triage hat er grosse Erfahrungen, da er schon mehrmals an Wiederholungskursen teilgenommen hat und vertraut ist mit den Abläufen. Weiter besitzt er Führungskompetenzen aus dem zivilen Leben, die er sich als Lehrmeister, Teamleader sowie aus mehreren Kursen angeeignet hat. Sehr praktisch sind auch seine guten Verbindungen mit dem höheren Kadern, dies ermöglicht eine effizientere Kommunikation.
Pain Points & Frustrationen	Zum Teil fühlt sich Iwo ein bisschen frustriert, wenn seine zugewiesenen Soldaten nicht motiviert sind und die Arbeitsleistungen ungenügend sind. Problematisch sind auch Meldezettel, die nicht vollständig ausgefüllt werden oder gar verloren gehen. Ab und zu kommt es auch vor, dass Meldungen lange unterwegs sind bis sie das Ziel erreichen, oder nicht ganz klar ist wer dafür zuständig ist. Dies kann so weit führen, dass Meldungen wieder zurück zur Triage kommen, ohne den korrekten Empfänger wirklich zu erreichen. (Verweis Interview 2 Probleme bei der Arbeit)
Eigenschaften & Behaviour Variables	Technisches Knowhow, Motivation während Arbeit, Zufriedenheit mit aktuellem Ablauf, Effizienz bei der Arbeit, Erfahrungen in diesem Bereich, Regelmässigkeit der Verwendung, Genauigkeit/Ausführlichkeit Meldezettel, Militärisches/Strategisches Denken, Auffassungsgabe, Fachliche Kompetenz (Eist Tm)
Ziele	Effizienz der Meldungen verbessern, Verlieren von Meldezetteln vermeiden, Abläufe optimieren, Alle notwendigen Informationen auf Zettel, Kommunikation im Team verbessern



4.5.2.1 Day-in-the-Live Szenario

Iwo ist ein Morgenmensch deshalb hat er kein Problem bereits um 06:45 Uhr aufzustehen. Nach der obligaten Dusche und dem üblichen Morgenritual (Gesicht waschen, Gang auf Toilette etc.) begibt er sich zum Frühstück. Diese Stärkung ist ihm sehr wichtig, deshalb isst er ausgiebig. Pünktlich um 07:30 Uhr betritt er den Kommandoposten und schaut sich kurz die anstehenden Arbeiten für den aktuellen Tag an. Es handelt sich um ein dicht gedrängtes Programm, da derzeit Übungen durchgeführt werden, bei denen viele Personen benötigt werden. Zuerst überblickt er kurz den gegenwärtigen Arbeitsplan, auf Anhieb sieht er, dass einzelne Mitarbeiter unterbeschäftigt sind. Aus diesem Grund teilt er zwei Personen weiteren Arbeiten zu, dadurch können Kapazitäten freigeräumt werden, die für andere Arbeiten dringend benötigt werden. Nach diesen Anpassungen druckt er den neuen Plan aus und teilt dies den betroffenen Soldaten mit. Die Soldaten nehmen es ein bisschen mürrisch zu Kenntnis, da dies zu zusätzlichem Arbeitsaufwand führt. Nichts desto trotz wird der neue Plan eingesetzt. Iwo fällt zudem auf, dass an einem Arbeitsplatz ein Soldat fehlt. Er erfährt durch einen anderen AdA, dass er heute Morgen noch nicht aufgetaucht ist. Deshalb beschliesst er ihn selber schnell aufzuwecken.

Für den Morgen stehen noch weitere kleinere Büroarbeiten an, die nicht weiter im Detail erwähnt werden. Vor dem Mittag gibt es noch eine kurze KP-Sitzung mit dem höheren Kader, indem die höheren Offiziere über die anstehende Übung informieren. Iwo macht sich pflichtbewusst einige Notizen zu den Dingen, die ihn sowie sein Team betreffen. Anschliessend macht er sich auf um pünktlich beim Mittagessen zu erscheinen. Über den Mittag bleibt noch genügend Zeit um sich kurz über die anstehenden Arbeiten, sowie einige private Sachen auszutauschen. Nach diesem kurzen Unterbruch arbeitet Iwo weiter an den anstehenden Aufgaben, bis ganz plötzlich ein höherer Offizier hereinplatzt und ihn unterbricht. Er müsse sofort die Personen neu einteilen, da anscheinend neue Aufträge eingegangen sind und dies eine neue Planung des Personals erfordert. Wie gefordert macht sich nun Iwo an die Arbeit und erarbeitet einen neuen Schichtplan. Dies fällt ihm jedoch nicht ganz einfach, da viele Arbeiten anstehen und verhältnismässig nur wenig Personal zur Verfügung steht. Trotzdem nimmt er sich dieser Aufgabe an, nach einiger Zeit kann er den neuen Plan ausdrucken und allen involvierten Personen zeigen. Iwo braucht mal eine Pause, deshalb geht er kurz an die frische Luft, trinkt etwas und diskutiert mit einigen Kameraden über verschiedene oberflächliche Themen. Beim Zurückkehren in die ZSA wird er aufgrund von vielen eingehenden Telefonen dringend in der Triage erwartet. Deshalb hilft er der Mannschaft bis zum Abendessen mit der Entgegennahme von Telefonaten aus. Zurzeit herrscht ein wirklich ausgesprochen hektisches Treiben, es gehen sehr viele Telefone ein, aus diesem Grund ist es wichtig, dass keine Meldungen verloren gehen. Nach unzähligen Telefonen ist es bald 18:00 Uhr, sein direkter Vorgesetzter - ein erfahrener Offizier, entlässt ihn deshalb und wünscht ihm viel Spass im FAK Ausgang. Für den FAK Ausgang muss sich Iwo noch kurz bereit machen, indem er den vorgesehen "Ausgänger" anzieht und an das AV (Abendverlesen) geht.

Den restlichen Abend verbringt er weitgehend in der Dorfkneipe mit Kameraden, dabei spielt er diverse Kartenspiele. Die Zeit vergeht rasend schnell, wobei er achten muss, dass er pünktlich wieder zurück in der ZSA ist. Zurück in der ZSA begibt er sich um ca. 12 Uhr zu Bett.



4.5.2.2 Ist-Szenarien

Ist-Szenario 1 Personal einteilen

Ausgangssituation Iwo hat eine fest definierte personelle Kapazität, die er best möglich Einsetzen sollte. Er beginnt mit der Planung der Mitarbeiter, als plötzlich neue Aufträge eingehen.

Auslöser Es gehen neue Aufträge ein, personelle Ressourcen sind verhindert

Schritte Um dies bestmöglich zu meistern, hat er eine Planungsübersicht mit einer Matrix, in der er eine Auswahl von verfügbaren Personen sowie den derzeitigen Aufträgen sieht. Anhand von dieser Matrix erstellt er einen Schichtplan und weist die einzelnen Personen bestimmten Arbeiten zu.

Probleme In der Triage werden aufgrund von der persönlichen Überbringungen der Meldungen zusätzliche Ressourcen benötigt

Ziele Einen guten Plan zu erstellen, der gerecht und fair ist sowie möglichst die Fähigkeiten der einzelnen Mitarbeiter beachtet, Kapazitäten effizient zu nutzen

4.5.2.3 Soll-Szenarien

Soll-Szenario 1 Ressourcen sparen

Ausgangssituation Iwo hat eine fest definierte personelle Kapazität, die er beste möglich Einsetzen sollte. Er beginnt mit der Planung der Mitarbeiter, als plötzlich neue Aufträge eingehen.

Auslöser Es gehen neue Aufträge ein, personelle Ressourcen sind verhindert

Schritte Aufgrund der eingesetzten Applikation EistCockpit müssen die Triage Mitarbeiter die Meldungen nicht mehr selber überbringen. Dadurch können massgeblich Ressourcen gespart werden, die für andere Arbeiten wiederum eingesetzt werden können. Aus diesem Grund teilt er für die Triage einen Mann weniger ein, welchen er stattdessen für die Abteilung "Richtstrahl Planer" verwenden kann.

Probleme Eventuell sind die Ressourcen zu knapp um alle Arbeiten ordnungsgemäss zu erfüllen. (Verweis Interview 2 Probleme bei der Arbeit)

Ziele Effizienterer Einsatz von verfügbaren Ressourcen, Planung optimieren



4.5.3 Persona Viktor Versiert



Abbildung 25: Persona Viktor Versiert

Alter	*30 Jahre*
Zivilstand	*Verheiratet, ein Kind*
Ausbildung	*Gymnasium, Master of marketing management, Projektleiter*
Tätigkeit	*Arbeitet für ein grosses Unternehmen in der Marketing Abteilung*

Technische Kenntnisse	Führungsqualitäten, gute rhetorische Fähigkeiten, Anwenderkenntnisse im Umgang mit Computer
Funktion	Viktor hat den Rang Hauptmann und ist Leiter der Abteilung Operationen und schlussendlich der Oberste Entscheidungsträger in seinem Gebiet. Er ist auf ein starkes und eingespieltes Team angewiesen und muss jederzeit mit den neusten Informationen versorgt werden.
Arbeitskontext	Der primäre Arbeitsplatz von Viktor ist die Eist, die Sekundäre ein eher ruhigeres, kleines Büro. In der Eist werden alle Informationen zusammengetragen und auf den Führungswänden festgehalten und aktualisiert. Von hier aus fällt Viktor seine Entscheidungen und gibt die Befehle aus. In seinem Büro arbeitet er an formalen Befehlen und Präsentationen für die Befehlsausgabe.
Arbeitsstil	Viktor ist erfahren und engagiert. Er weiss sein Team zu leiten, Aufgaben zu delegieren und in kritischen Situationen einen kühlen Kopf zu bewahren. Er ist gut organisiert und kann der Situation entsprechen Prioritäten setzen. Allerdings ist er dafür auch auf den Informationsfluss von aussen angewiesen. Seine Mitarbeiter führt er bestimmt aber stets mit Vorbild an.
Persönlichkeit & Vorlieben	Viktor ist ein Selbstbewusster Mann, der weiss was er will und was seine Vorgesetzten von ihm erwarten. Er definiert klar was er von seinen Unterstellten erwartet und fordert dies auch konsequent ein. Privat ist Viktor ein aufgestellter Optimist mit ein wenig Lebenserfahrung, hat eine Frau und ein Kind im Primarschulalter. Er hat zwar eine Vorliebe für schnelle Autos, fährt aber einen Skoda Kombi. Viktor ist ausserdem bei der Feuerwehr engagiert.
Vorkenntnisse & Lernen	Viktor besitzt sehr gute Anwenderkenntnisse im Umgang mit dem Computer, kann triviale Probleme auch selber lösen. Die Arbeitsprozesse seiner Abteilung kennt er wie kein Zweiter, zumal er sie teilweise mitgestaltet und optimiert hat.



Pain Points & Frustrationen Viele Informationen gelangen nur unvollständig bis gar nicht zu ihm, was seine Aufgabe als Entscheidungsträger erschwert. Die Angeschlossenen Aussenposten und Knoten melden sich nicht wie gefordert rechtzeitig. Daher kommt es oft vor, das Viktor den Infos nachrennen muss. Ausserdem fehlt auf den eingegangenen Meldungen eine Priorisierungsmöglichkeit. (Verweis Interview 4 Probleme bei der Arbeit)

Eigenschaften & Behaviour Variables Technisches Knowhow, Motivation während Arbeit, Effizienz bei der Arbeit, Erfahrungen in diesem Bereich, Regelmässigkeit der Verwendung, Genauigkeit/Ausführlichkeit Meldezettel, Militärisches/Strategisches Denken, Auffassungsgabe, Fachliche Kompetenz (Eist Tm)

Ziele Effizienz der Meldungen verbessern, Verlieren von Meldezetteln vermeiden, Abläufe optimieren, Alle notwendigen Informationen auf Zettel, Kommunikation im Team verbessern

4.5.3.1 Ist-Szenarien

Ist-Szenario 1 Uninformierte Mitarbeiter/Falsche Infos:

Ausgangssituation Viktor plant in seinem Büro gerade die anstehenden Inspektionen der Staos, als ein Mitarbeiter der Triage an die Türe klopft und eintritt. Er übergibt Viktor einen Meldezettel der besagt, einem Stao gehe der Treibstoff aus und sie bräuchten dringend Nachschub. Auf dem Meldezettel ist jedoch kein Stao erwähnt, lediglich ein schwer leserlicher Name: Wm Gewehla.

Auslöser Anruf eines Stao-Chefs an die Triage

Schritte Viktor bringt den Meldezettel sofort in die Triage zurück und fragt nach dem Stao. Der Mitarbeiter hat aber keine Ahnung und gibt an nicht gewusst zu haben, dass man das nachfragen muss. In den Standortlisten findet sich keine Person mit diesem Namen, offenbar hat der zuständige Mitarbeiter den Namen nicht richtig verstanden und auch nicht nachgefragt. Erst nach einigem Suchen kommt der Name von "Wm Andi Gewehre" vom Stao 3 zum Vorschein. Viktor bestellt den Nachschub

Probleme Uninformierte Mitarbeiter (Kennen die zu stellenden Fragen nicht), Falsche Infos

Ziele Meldungen werden immer vollständig erfasst, Mitarbeiter sind besser informiert, System leitet die Mitarbeiter durch den Prozess



Ist-Szenario 2 An Personen gebundene Meldungen:

Ausgangssituation Viktor ist gerade auf einer Stao-Inspektion, als ein Mitarbeiter der Triage ihm einen Meldezettel zur schnellen Bearbeitung überbringen will (Die Zuständigkeit hat er dem Telefonat entnommen). Der Meldezettel besagt dass Stao 3 demnächst über keinen Treibstoff mehr verfügt und Nachschub braucht. Da Viktor nicht da ist, legt der Mitarbeiter ihm den Zettel in sein Posteingangsfach

Auslöser Anruf eines Stao-Chefs an die Triage

Schritte Viktor ist für die nächsten 48h auf Inspektion und hat Alle seine Kompetenzen dem Zellenchef Stv., Oblt Bruno Brenzlig, übergeben. Der Zettel bleibt liegen bis Viktor wieder in Eist ist, doch dann ist es bereits zu spät; Die Verbindung zum Stao 3 ist abgebrochen.

Probleme Meldungen gehen verloren, Meldungen können nur an 1 Person geschickt werden

Ziele Meldungen können an Gruppen und Funktionen übermittelt werden anstatt nur an 1 Person, Meldungen kommen Zeitgerecht an

Ist-Szenario 3 Bidirektionale Meldungen, Feedback erledigt/Nicht erledigt:

Ausgangssituation Viktor plant in seinem Büro gerade die anstehenden Inspektionen der Staos, als ein Mitarbeiter der Triage an die Türe klopft und eintritt. Er übergibt Viktor einen Meldezettel mit einem Antrag von Stao 2, die ihre Schüssel aufgrund der Stromversorgung um 50m verschieben würden.

Auslöser Anruf eines Stao-Chefs an die Triage

Schritte Viktor begibt sich in die Triage und verlautet, dass dem Antrag von Stao 2 stattgegeben ist, diese sollen doch so schnell wie möglich informiert werden und Rückmeldung geben, sobald das erledigt ist. (Verweis Interview 4 Wie reagieren Sie typischerweise auf eine Meldung? Ablauf?)

Probleme Die aus einer Meldung entstehenden Aufträge müssen mündlich verteilt werden, es gibt keine Erledigungskontrolle

Ziele Meldungen Bi-direktional, können weitervermittelt/erweitert werden, Feedback erledigt/nicht erledigt
(Verweis Interview 4 Was für Verbesserungswünsche hätten Sie am System?)



4.5.3.2 Soll-Szenarien

Soll-Szenario 1 Geleitete, uninformierte Mitarbeiter/Keine falschen Infos:

Ausgangssituation	Viktor plant in seinem Büro gerade die anstehenden Inspektionen der Staos. Als er fertig ist sieht er auf dem Bildschirm eine Meldung von Stao 3 die besagt, der Treibstoff gehe aus und sie bräuchten dringend Nachschub.
Auslöser	Anruf eines Stao-Chefs an die Triage
Schritte	Viktor bestellt den Nachschub bei der Logistik-Zelle und markiert die Meldung als erledigt
Probleme	Uninformierte Mitarbeiter (Kennen die zu stellenden Fragen nicht), Falsche Infos
Ziele	Meldungen werden immer vollständig erfasst, Mitarbeiter sind besser informiert, System leitet die Mitarbeiter durch den Prozess, Feedback erledigt, nicht erledigt

Soll-Szenario 2 An Funktionen/Zellen gebundene Meldungen:

Ausgangssituation	Viktor ist gerade auf einer Stao-Inspektion als eine Meldung über die Triage hereinkommt die besagt, der Treibstoff gehe aus und sie bräuchten dringend Nachschub.
Auslöser	Anruf eines Stao-Chefs an die Triage
Schritte	Bruno Brenzlig, der Zellenchef Stv. der während Viktors Abwesenheit über seine gesamten Kompetenzen verfügt, sieht die Meldung und bestellt unverzüglich Nachschub bei der Logistik-Zelle
Probleme	Meldungen gehen verloren, Meldungen können nur an 1 Person geschickt werden
Ziele	Meldungen können an Gruppen und Funktionen übermittelt werden anstatt nur an 1 Person, Meldungen kommen Zeitgerecht an (Verweis Interview 4 Was für Verbesserungswünsche hätten Sie am System?)



Soll-Szenario 3 Bidirektionale Meldungen, Feedback erledigt/Nicht erledigt:

Ausgangssituation	Viktor plant in seinem Büro gerade die anstehenden Inspektionen der Staos, als gerade eine Meldung mit einem Antrag von Stao 2, die ihre Schüssel aufgrund der Stromversorgung um 50m verschieben würden, hereinkommt.
Auslöser	Anruf eines Stao-Chefs an die Triage
Schritte	Viktor kommentiert den Antrag mit "Stattgegeben, Stao sofort informieren" und ändert die Zuständigkeit auf "Triage". Diese Informiert den Stao unverzüglich und setzt die Meldung auf erledigt.
Probleme	Die aus einer Meldung entstehenden Aufträge müssen mündlich verteilt werden, es gibt keine Erledigungskontrolle
Ziele	Meldungen Bi-direktional, können weitervermittelt/erweitert werden, Feedback erledigt/nicht erledigt (Verweis Interview 4 Was für Verbesserungswünsche hätten Sie am System?)



EistCockpit

5 Anforderungen

David Schöttl, Remo Waltenspül & Diego Steiner



5.1 Dokumenteninformation

Datum	Version	Änderung	Autor
06.06.2012	1.0	Erste Version des Dokuments	dsteiner
07.06.2012	1.1	Nicht funktionale Anforderungen dokumentiert	rwaltens
07.06.2012	1.2	Design Constraints dokumentiert	dsteiner
08.06.2012	1.3	Review	dschöttl



5.2 Allgemein

5.2.1 Zweck des Dokumentes

Dieses Dokument beschreibt die Funktionalen sowie die nichtfunktionalen Anforderungen an das Projekt EistCockpit.

5.2.2 Gültigkeitsbereich

Dieses Dokument gilt als Grundlage des Projektes und ist daher über die gesamte Projektdauer gültig (20.02 bis 07.06.2012).

5.2.3 Definitionen und Abkürzungen

Siehe „Glossar".



5.3 Tools

In der Nachfolgenden Tabelle sind die für das EistCockpit verwendeten Tools mit der Versionsnummer aufgelistet.

Tool	Version	Link	Bemerkungen
Visual Studio	10.0	http://www.microsoft.com/visualstudio/11/de-de	IDE
.net Framework	4.0	http://www.microsoft.com/germany/net/net-framework-4.aspx	
ReSharper	6.1.1	http://www.jetbrains.com/resharper/	Refactorings
Microsoft Office	2010	http://office.microsoft.com/de-ch/	Dokumentation
GIMP	2.8	http://www.gimp.org/	Bildbearbeitung
Doxygen	1.8.1	http://www.stack.nl/~dimitri/doxygen/	Code-dokumentation
Visual Paradigm	9.0	http://www.visual-paradigm.com/product/vpuml/	UML Diagramme
Enterprise Architect	9.0	http://www.sparxsystems.com.au/	UML Diagramme
Redmine	1.1.3	http://www.redmine.org/	Projekt-management
SVN	1.7	http://subversion.apache.org/	Versions-management
OmniGraffle	5.4	http://www.omnigroup.com/products/omnigraffle/	Diagramme
Balsamiq	2.1.18	http://www.balsamiq.com/	Mockups
Brain	0.9 Alpha	http://de.wikipedia.org/w/index.php?title=Gehirn&oldid=104014075	

Tabelle 37 Auflistung benutzter Tools



5.4 Funktionale Anforderungen

Als Teil von Scrum werden die Funktionalen Anforderungen als User Stories aufgelistet. Die User Stories sind mit dem ihnen zugewiesenen Sprint gekennzeichnet, in welchem dessen Umsetzung geplant ist. Sprints die mit (Klammern) gekennzeichnet sind haben zu bedeuten, dass in diesem Sprint ein Prototyp der Umsetzung erfolgt.

Userstory	Definition of Done	Sprint
Als Administrator kann ich neue oder erweiterte Funktionalität einfach einfügen	Plugins werden aus einem Ordner geladen und im dafür vorgesehen Bereich angezeigt	(1), 2
Als Administrator kann ich Daten einfach zwischen mehreren Dienstleistungen migrieren	Die Datenbank ist in einem einzigen File gespeichert und kann beliebig ausgetauscht werden	(1), 2
Als Benutzer ist es mir möglich gleichzeitig mit anderen Benutzern zu arbeiten	Die Applikation ist in mehrere Tiers aufgeteilt; Client und Server.	(1), 2
Als Benutzer ist es mir möglich eine Dienstleistung als Arbeitskontext auszuwählen	Im dafür vorgesehenen Bereich werden die Dienstleistungen angezeigt und können ausgewählt werden	2
Als Benutzer sehe ich welche die aktuell ausgewählte Dienstleistung ist	In der Statusbar wird die aktuell ausgewählte Dienstleistung angezeigt	2
Als Benutzer ist es mir möglich eine Operation als Arbeitskontext auszuwählen	Im dafür vorgesehenen Bereich nach Dienstleistungen gruppiert werden die Operationen angezeigt und können ausgewählt werden	2
Als Benutzer sehe ich welche die aktuell ausgewählte Operation ist	In der Statusbar wird die aktuell ausgewählte Operation angezeigt	2
Als Benutzer in der Eist sehe ich eine Übersicht der eingegangenen Meldungen	Im Meldungsplugin wird eine Tabelle mit den Meldungen angezeigt	2
Als Benutzer in der Eist kann ich Meldungen nach allen Kriterien sortieren	Im Meldungsplugin sind alle Spalten der Übersicht sortierbar	3
Als Benutzer in der Eist kann ich auf den ersten Blick sehen, welche Meldungen wichtiger sind als Andere	Im Meldungsplugin sind die Meldungen ja nach Priorität mit einer Farbe hinterlegt	3
Als Benutzer in der Eist kann ich Meldungen nach allen Kriterien filtern	Im Meldungsplugin ist für jede Spalte mindestens ein Filter auswählbar	2
Als Benutzer in der Eist sehe ich wenn jemand Anders eine neue Meldung in meinem Kriterienscope erfasst	Das Meldungsplugin aktualisiert regelmässig die lokalen Meldungen	3
Als Benutzer in der Eist kann ich die Filterkriterien ausblenden, um mehr Platz für Meldungen zu erhalten	Im Meldungsplugin können die Filterkriterien über einen Knopf ein- und ausgeblendet werden	2
Als Benutzer in der Triage kann ich einfach eine neue Meldung erfassen	Im Meldungsplugin ist der Knopf zum Erfassen einer Meldung gut sichtbar platziert. Die Meldung kann gespeichert werden	2
Als Benutzer in der Triage werde ich durch den Prozess des Erfassens geleitet, wenn eine neue Meldung	Im Fenster zur Meldungserfassung ist Schritt für Schritt beschrieben, was abgefragt werden muss	2



hereinkommt		
Als Benutzer in der Triage kann ich eine Meldung die noch nicht fertig ist abspeichern ohne sie abzusenden	Im Fenster zur Meldungserfassung gibt es einen Knopf um die Meldung als Entwurf abzuspeichern	3
Als Benutzer in der Triage kann ich erfasste Meldungen bearbeiten	Im Meldungsplugin können bereits erfasste Meldungen geöffnet und bearbeitet werden	2
Als Benutzer in der Triage sehe ich wenn ich eine Meldung abgeschickt habe das Diese versandt wurde	Nachdem der Senden-Knopf im Fenster zu Meldungserfassung gedrückt und die Meldung versandt wurde wird in der Statusleiste eine entsprechende Meldung angezeigt	2
Als Benutzer in der Eist kann ich die zu einer Operation erfassten Standorte anzeigen	Im Standorteplugin wird eine Übersicht der Standorte zur aktuellen Operation angezeigt	2
Als Benutzer in der Eist kann ich Details zu einem Standort anzeigen	Im Standorteplugin werden die Informationen zu einem ausgewählten Standort angezeigt	2
Als Benutzer in der Eist kann ich neue Standorte anlegen	Im Standorteplugin kann ein neuer Standort hinzugefügt werden	2
Als Benutzer in der Eist kann ich vorhandene Standorte bearbeiten	Im Standorteplugin kann ein Standort bearbeitet werden	2
Als Benutzer in der Eist kann ich vorhandene Standorte löschen	Im Standorteplugin kann ein Standort gelöscht werden	2
Als Benutzer in der Eist kann ich keine ungültigen Angaben zu einem Standort machen	Im Standorteplugin werden sämtliche Eingaben überprüft, sobald Änderungen vorliegen	3
Als Administrator kann ich neue Dienstleistungen hinzufügen	Im Dienstleistungsplugin kann eine neue Dienstleistung hinzufügen	3
Als Administrator kann ich Dienstleistungen bearbeiten	Im Dienstleistungsplugin kann eine Dienstleistung bearbeiten	3
Als Administrator kann ich Dienstleistungen löschen	Im Dienstleistungsplugin kann eine Dienstleistung löschen	3
Als Administrator kann ich neue Operation hinzufügen	Im Dienstleistungsplugin kann eine neue Operation hinzufügen	3
Als Administrator kann ich Operation bearbeiten	Im Dienstleistungsplugin kann eine Operation bearbeiten	3
Als Administrator kann ich Operation löschen	Im Dienstleistungsplugin kann eine Operation löschen	3
Als Administrator kann ich Plugins für normale Benutzer sperren	Im Konfigurationsplugin kann definiert werden ob ein Plugin Administratorenrechte benötigt	3
Als Administrator kann ich festlegen, welche Personen über Administratorenrechte verfügen	Im Konfigurationsplugin kann definiert werden, welche ActiveDirectory-Rollen in der Applikation Administratorenrechte haben	3
Als Entwickler des EistCockpits kann ich sehen, welchen Zweck eine Klass/Methode verfolgt	Die Codedokumentation ist für jede Klasse generiert	3

Tabelle 38 Auflistung funktionaler Anforderungen



5.5 Nichtfunktionale Anforderungen

Die nachfolgenden nichtfunktionalen Anforderungen definieren Anforderungen, an die „Qualität in welcher die geforderte Funktionalität zu erbringen ist.[4] Dabei können zum Teil nicht alle Anforderungen unmittelbar aus den User Stories abgeleitet werden, aus diesem Grund werden diese in dem folgenden Abschnitt festgehalten.

5.5.1 Funktionalität

5.5.1.1 Angemessenheit

Die Bedürfnisse der verschiedenen Anwender sind sehr unterschiedlich. Aus diesem Grund müssen die Funktionen benutzergerecht und intuitiv zu benutzen sein. Wichtig ist auch, dass die am häufigsten benutzten Funktionen schnell zu finden sind und nicht mit vielen weniger oft verwendeten Funktionen überladen werden. Das erfüllen der Angemessenheit soll mittels eines Usability-Tests verifiziert werden. Dieser Test wird jedoch erst in der folgenden Bachelorarbeit durchgeführt.

5.5.2 Zuverlässigkeit

5.5.2.1 Reife

Die Applikation sollte nach Beendigung dieser Studienarbeit einen guten Reifegrad für die implementierten Funktionen haben. Um dies sicherzustellen sollen die Kernfunktionen mit automatischen Unit Tests getestet werden. Wobei die durchschnittliche Testabdeckung für die Plugins über 70% und für den Data Access Layer über 90% liegen sollte.

5.5.3 Benutzbarkeit

5.5.3.1 Verständlichkeit & Erlernbarkeit

Da die Benutzer unterschiedliches technisches Vorwissen mitbringen, und zum Teil auch weniger Technik affine Anwender die Applikation einsetzen werden, ist eine wichtige Anforderung, dass der Einstieg sowie die einfache Erlernbarkeit sehr schnell möglich ist. Es soll nicht notwendig sein bei der erstmaligen Nutzung der Anwendung ein Benutzerhandbuch aufzuschlagen. Das System soll mit dem Benutzer interagieren und ihn durch Hinweise unterstützen seine Aufgaben erfolgreich durchzuführen.

5.5.3.2 Bedienbarkeit

Die Benutzer haben nicht lange Zeit um sich mit der Applikation auseinanderzusetzen, zudem sind die Computerkenntnisse sehr unterschiedlich. Deshalb muss die Bedienbarkeit intuitiv, einfach und zielführend sein. Um dies zu erreichen, sollen die am häufigsten verwendeten Funktionen hervorgehoben werden und gut erreichbar sein. Das System soll dem Benutzer ein unterstützendes Gefühl vermitteln, und ihn nicht mit erweiterten Funktionalitäten überfordern.

Um diese geforderten Anforderungen zu überprüfen wird in der folgenden Bachelorarbeit ein Usability Test durchgeführt.

---

[4] [heini12]: Funktionale, nicht funktionale Anforderungen, http://www.anforderungsmanagement.ch/in_depth_vertiefung/funktionale_nicht_funktionale_anforderungen/ (06.06.2012)



5.5.3.3 Attraktivität

Die Attraktivität spielt keine zentrale Rolle, trotzdem soll sich das äussere Erscheinungsbild an dem Stil der vorhandenen Applikationen der Schweizer Armee anlehnen.

5.5.4 Effizienz

5.5.4.1 Zeitverhalten

In hektischen Situationen gehen viele Anrufe ein, die in kurzer Zeit abgearbeitet werden müssen. Daher ist es wichtig, dass Meldungen innerhalb von einer Minute erfasst werden können.

5.5.5 Änderbarkeit & Wartbarkeit

5.5.5.1 Analysierbarkeit

Der Code sollte vollständig dokumentiert sein. Anhand des dokumentierten Codes soll es möglich sein die API automatisch zu generieren. Folge dessen sollte der Code mittels der Code-Dokumentation schnell zu verstehen und allfällige Fehlerquellen einfach zu identifizieren sein.

5.5.5.2 Modifizierbarkeit

Eine grundlegende Anforderung ist es, dass die Software einfach erweitert, modifiziert werden kann. Um diese Anforderung zu erfüllen, soll das System auf einer Plugin Architektur aufsetzen. Zudem sollen Programmteile, bei denen eine grosse Änderungswahrscheinlichkeit besteht, möglichst nur an einem Ort implementiert werden



5.6 Design Constraints

Die Schweizer Armee beschränkt die IT-Mittel innerhalb der Eist auf das TEPLAS. Dieses setzt sich aus einem Domänencontroller, einem Citrix-Server, mehreren Windows XP Clients und Druckern zusammen. Diese Umgebung bildet die Rahmenbedingungen für das EistCockpit; Es dürfen keine zusätzlichen Hardwarekomponenten verwendet werden.



5.7 Zugänglichkeit (Accessibility)

Laut Major Hans-Andrea Veraguth sind in der Schweizer Armee keine AdAs mit schwerwiegenden Wahrnehmungsstörungen zugelassen (Siehe Anhang Sitzungsprotokoll vom 5. März, Abend). Es ist also keine Anforderung, EistCockpit auf besondere Bedürfnisse auszulegen. Es können jedoch Fälle von Rot-Grün Schwäche vorkommen, weshalb bei Status die Farbe nicht der einzige Indikator sein darf.



EistCockpit

6 Domain Analyse

David Schöttl, Remo Waltenspül & Diego Steiner



6.1 Dokumenteninformation

Datum	Version	Änderung	Autor
05.06.2012	1.0	Erste Version des Dokuments	dsteiner
06.06.2012	1.1	Dokumentation Datenmodel	dsteiner
07.06.2012	1.2	UI Design Entscheide	rwaltens
08.06.2012	1.3	Review	rwaltens



6.2 Allgemein

6.2.1 Zweck des Dokumentes

Dieses Dokument beschreibt das Domainmodell, die Prozesse innerhalb der Einheit sowie das daraus geschlossene Datenmodell für das EistCockpit. In einem zweiten Teil wird auf das Userinterface, Designentscheidungen sowie auf die Designguides eingegangen.

6.2.2 Gültigkeitsbereich

Dieses Dokument gilt als Grundlage des Projektes und ist daher über die gesamte Projektdauer gültig (20.02 bis 07.06.2012).

6.2.3 Definitionen und Abkürzungen

Siehe „Glossar“.



6.3 Domainmodell

Nachfolgend ist das gesamte Domainmodell dargestellt.

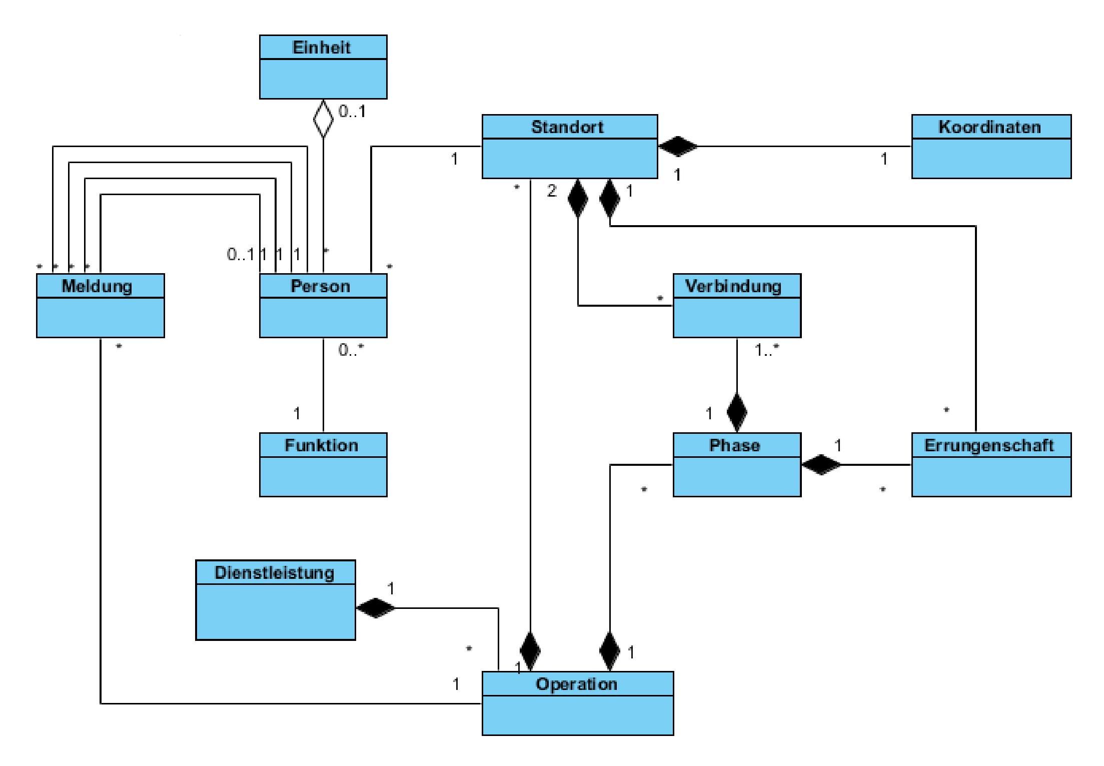


Abbildung 26: Domain Modell

6.3.1.1 Dienstleistung

Das Einsatzgebiet für das EistCockpit ist primär die Einsatzstelle Telematik innerhalb eines der jährlichen Wiederholungskurse einer WK-Einheit. Weil die Daten nicht WK-übergreifend verfügbar sein müssen, ist als Grundrahmen für die Daten jeweils eine Dienstleistung vorgesehen.

6.3.1.2 Operation

Innerhalb der Dienstleistung sind die Aktivitäten in Operationen aufgeteilt, die über eine vordefinierte Zeitspanne während der Dienstleistung stattfinden. Eine Operation kann in verschiedenen Arten daher kommen; Es kann eine Übung, ein Einsatz oder

6.3.1.3 Standort

Um den Auftrag „Betreiben der Kommunikationsnetze der Armee“ erfüllen zu können wird das Personal für die Dauer einer Operation auf verschiedene Standorte aufgeteilt.

6.3.1.4 Meldung

Das Augenmerk liegt auf den Meldungen die über die Triage in die Einsatzstelle Telematik hereinkommen. Sie beinhalten Grundlegende Statusveränderungen wie das Erreichen oder Verlassen des Standortes, aber auch Rückfragen oder Bestellungen logistischer Natur.Diese Meldungen werden in der Triage erfasst und dann an die korrekte Stelle weitergeleitet.



6.4 Prozesse

6.4.1 Dienstleistungen

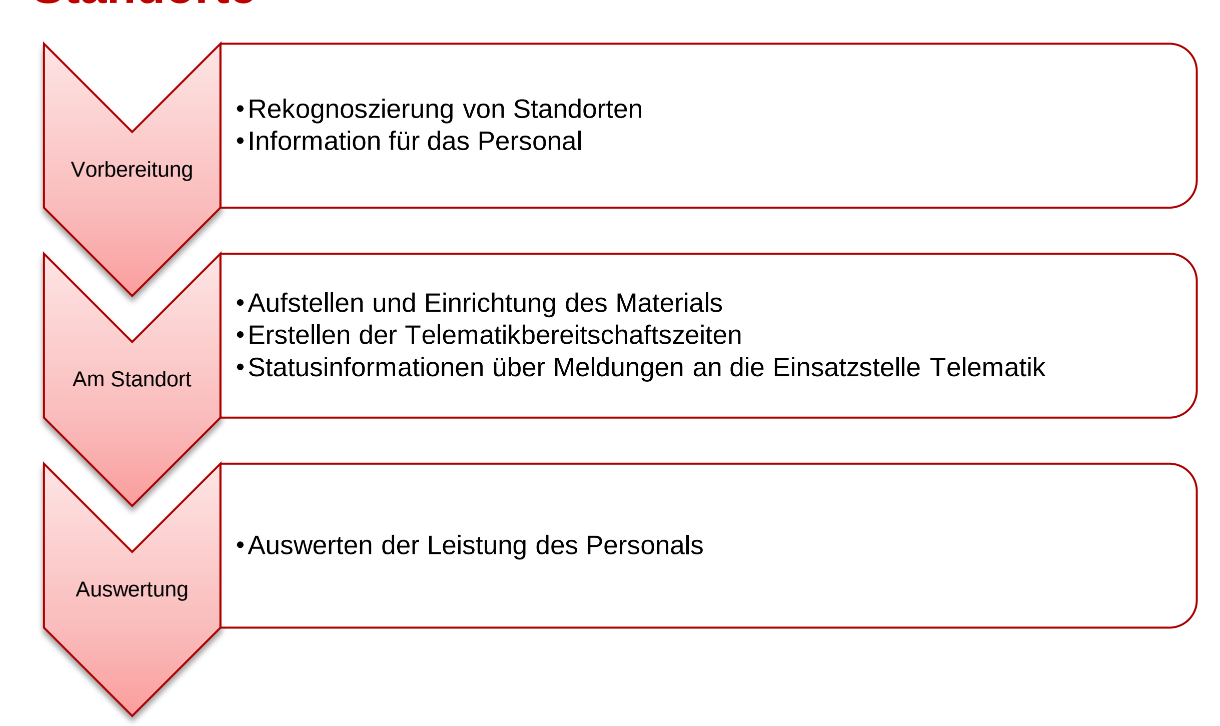
Planung	• Festlegung der Zeitraumes, Ort und Ziel • Einteilung von Personal und Material • Planung von Operationen
Vorbereitung	• Rekognoszierung von Durchführungsorten • Dienstleistungsaufforderung für das Personal • Bestellung von Fahrzeugen, Material und Unterlagen
Ausführung	• Ausbildung des Personals • Erstellen des Tagesgeschäfts • Detailplanung und Durchführung von Operationen
Auswertung	• Auswerten der Leistung des Personals • Auswertung der Rahmenbedingungen als Faktor für die nächste Dienstleistung

6.4.2 Operationen

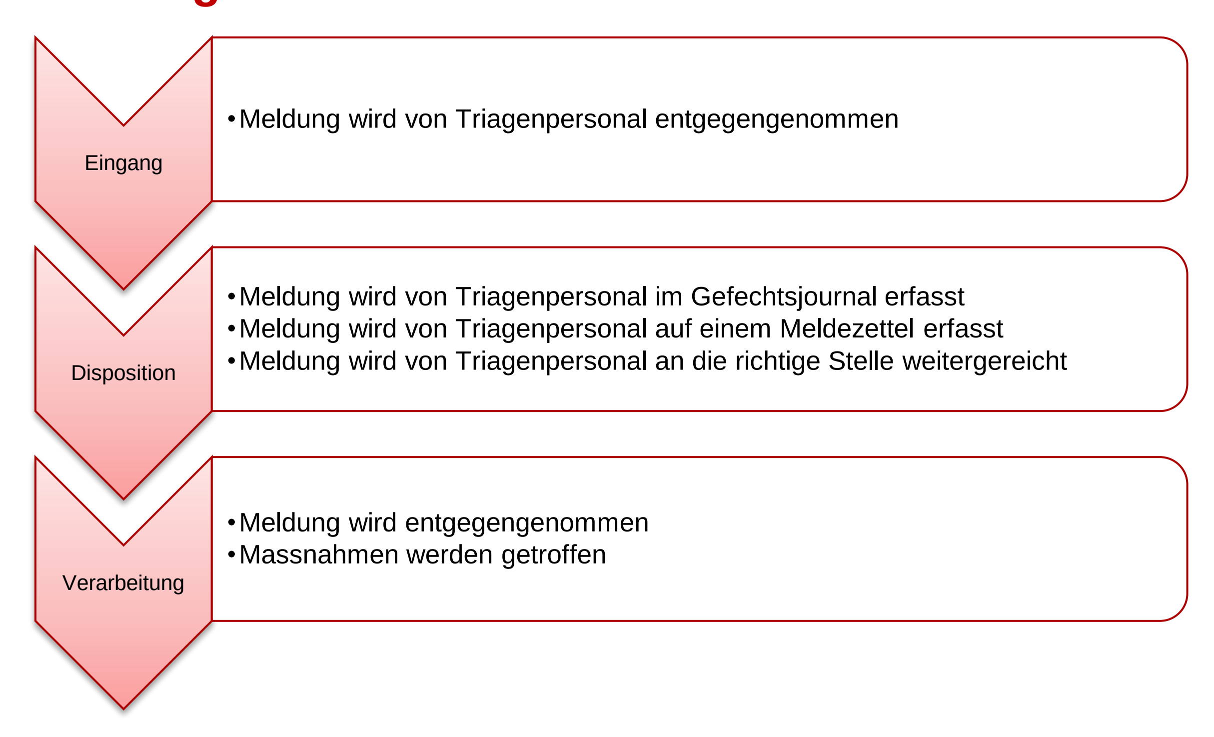
Planung	• Festlegung der Zeitraumes, Ort und Ziel • Einteilung von Personal und Material • Planung der Standorte
Vorbereitung	• Rekognoszierung von Standorten • Verfassung des Befehls • Information für das Personal
Ausführung	• Überwachung des Ablaufs • Überwachung der Standorte mittels Meldungen • Laufende Planung aufgrund der Überwachung
Auswertung	• Auswerten der Leistung des Personals



6.4.3 Standorte



6.4.4 Meldungen





6.5 Datenmodell

6.5.1 Dienstleistungen

Unit
Properties
id
name
parentID
Navigation Properties
Locations
Persons

Location
Properties
id
fullName
shortName
address
ownerAddress
ownerContact
personResponsibleID
unitID
coordinateID
contact
operationID
from
to
description
Navigation Properties
SlaveConnections
MasterConnections
Coordinate
Operation
Unit
ResponsiblePerson
LocationsPersons
Achievements

Coordinate
Properties
id
altitude
latitudeGPS
longitudeGPS
latitudeLV03
longitudeLV03
latitudeLV95
longitudeLV95
Navigation Properties
Location

Message
Properties
id
senderID
recipientID
priority
type
text
creatorID
created
modified
modifierID
operationID
isDraft
Navigation Properties
Recipient
Sender
Creator
Modifier
Operation

Person
Properties
id
rank
firstName
lastName
address
phone
remarks
status
unitID
functionID
Navigation Properties
LocationResponsibilities
LocationsPersons
Unit
ReceivedMessages
SentMessages
CreatedMessages
Function
ModifiedMessages

Connection
Properties
firstLocationID
secondLocationID
setupPhaseID
teardownPhaseID
Navigation Properties
MasterLocation
SlaveLocation
TeardownPhase
SetupPhase

Function
Properties
id
function
description
Navigation Properties
LocationsPersons
Persons

LocationPerson
Properties
locationID
personID
functionID
Navigation Properties
Location
Person
Function

Phase
Properties
id
name
description
order
operationID
from
to
Navigation Properties
TeardownConnections
SetupConnections
Operation
Achievements

Achievement
Properties
id
description
type
phaseID
locationID
due
achieved
Navigation Properties
Location
Phase

Service
Properties
id
name
description
from
to
Navigation Properties
Operations

Operation
Properties
id
name
type
serviceID
Navigation Properties
Locations
Service
Phases
Messages

AdminRole
Properties
ID
ADGroup
ConfigurationID
Navigation Properties
Configuration

Configuration
Properties
ID
PollInverval
Navigation Properties
AdminRoles
PluginAccesses

PluginAccess
Properties
ID
PluginName
RequireAdmin
ConfigurationID
Navigation Properties
Configuration

Abbildung 27: Datenmodell Service



6.5.1.1 Service

Repräsentiert die Einheit einer Dienstleistung.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identität
Name	*String*	Name der Dienstleistung; Bsp.: „WK 2011“
Description	*Text*	Beschreibung der Dienstleistung; Bsp.: „WK 2011 in Luzern“
From	*Datetime*	Beginn der Dienstleistung
To	*Datetime*	Ende der Dienstleistung
Operations	*Relation*	Die der Dienstleistung zugeordneten Operationen

Tabelle 39 Beschreibung Entität Service



6.5.2 Operationen

Unit
Properties
id
name
parentID
Navigation Properties
Locations
Persons

Location
Properties
id
fullName
shortName
address
ownerAddress
ownerContact
personResponsibleID
unitID
coordinateID
contact
operationID
from
to
description
Navigation Properties
SlaveConnections
MasterConnections
Coordinate
Operation
Unit
ResponsiblePerson
LocationsPersons
Achievements

Coordinate
Properties
id
altitude
latitudeGPS
longitudeGPS
latitudeLV03
longitudeLV03
latitudeLV95
longitudeLV95
Navigation Properties
Location

Message
Properties
id
senderID
recipientID
priority
type
text
creatorID
created
modified
modifiedID
operationID
isDraft
Navigation Properties
Recipient
Sender
Creator
Modifier
Operation

Person
Properties
id
rank
firstName
lastName
address
phone
remarks
status
unitID
functionID
Navigation Properties
LocationResponsibilities
LocationsPersons
Unit
ReceivedMessages
SentMessages
CreatedMessages
Function
ModifiedMessages

Connection
Properties
firstLocationID
secondLocationID
setupPhaseID
teardownPhaseID
Navigation Properties
MasterLocation
SlaveLocation
TeardownPhase
SetupPhase

Function
Properties
id
function
description
Navigation Properties
LocationsPersons
Persons

LocationPerson
Properties
locationID
personID
functionID
Navigation Properties
Location
Person
Function

Phase
Properties
id
name
description
order
operationID
from
to
Navigation Properties
TeardownConnections
SetupConnections
Operation
Achievements

Achievement
Properties
id
description
type
phaseID
locationID
due
achieved
Navigation Properties
Location
Phase

Service
Properties
id
name
description
from
to
Navigation Properties
Operations

Operation
Properties
id
name
type
serviceID
Navigation Properties
Locations
Service
Phases
Messages

AdminRole
Properties
ID
ADGroup
ConfigurationID
Navigation Properties
Configuration

Configuration
Properties
ID
PollInverval
Navigation Properties
AdminRoles
PluginAccesses

PluginAccess
Properties
ID
PluginName
RequireAdmin
ConfigurationID
Navigation Properties
Configuration

Abbildung 28: Datenmodell Operation



6.5.2.1 Operation

Repräsentiert eine Operation innerhalb einer Dienstleistung

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identität
Name	*String*	Name der Operation; Bsp.: „U ´SIMPLICITAS´“
Type	*Enum*	Der Typ der Operation
ServiceID	*Int*	Die ID der Dienstleistung, welcher die Operation zugeordnet ist
Locations	*Relation*	Die Standorte, die der Operation zugeordnet sind
Service	*Relation*	Die Dienstleistung, welcher die Operation zugeordnet ist
Phases	*Relation*	Die Phasen die für diese Operation definiert sind (TBZ)
Messages	*Relation*	Die Meldungen die während dieser Operation erfasst wurden

Tabelle 40 Beschreibung Entität Operation

Operationstypen

Eine Operation kann verschieden Type annehmen:

- Übung
- Einsatz



6.5.2.2 Phase

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identität
Name	*String*	Name der Phase; Bsp.: „ALPHA“
Description	*Text*	Beschreibung der Phase
Order	*Int*	Numerisches Gewicht für die Reihenfolge der Phasen
OperationID	*Int*	Die ID der Operation, der die Phase zugeordnet ist
From	*Datetime*	Beginn der Phase
To	*Datetime*	Ende der Phase
TeardownConnections	*Relation*	Die Verbindungen, die in dieser Phase abgebaut werden
SetupConnection	*Relation*	Die Verbindungen, die in dieser Phase aufgebaut werden
Operation	*Relation*	Die Operation, der die Phase zugeordnet ist
Achievments	*Relation*	Die Errungenschaften dieser Phase, Spezifischer die TBZ

Tabelle 41 Beschreibung Entität Phase



6.5.2.3 Achievment

Repräsentiert eine Errungenschaft die in einer Phase von einem Standort zu einem bestimmten Zeitpunkt erreicht werden muss.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identität
Description	*Text*	Beschreibung der Errungenschaft
Type	*Enum*	Typ der Errungenschaft
PhaseID	*Int*	Die ID der Phase, der die Errungenschaft zugeteilt ist
LocationID	*Int*	Die ID des Standortes, dem die Errungenschaft zugeteilt ist
Due	*Datetime*	Zeitpunkt, bis wann die Errungenschaft erreicht werden muss
Achieved	*Boolean*	Ist die Errungenschaft erreicht
Location	*Relation*	Der Standort, dem die Errungenschaft zugeordnet ist
Phase	*Relation*	Die Phase, der die Errungenschaft zugeteit ist

Tabelle 42 Beschreibung Entität Achievment

Errungenschaftstypen

Eine Errungenschaft kann folgende Typen annehmen:

- Bereitschaftsraum erreicht
- Bereitschaftsraum verlassen
- Standort erreicht
- Standort verlassen
- Telematikbereitschaftszeit SHF
- Telematikbereitschaftszeit IMFS



6.5.3 Standorte

Unit
Properties
id
name
parentID
Navigation Properties
Locations
Persons

Message
Properties
id
senderID
recipientID
priority
type
text
creatorID
created
modified
modifierID
operationID
isDraft
Navigation Properties
Recipient
Sender
Creator
Modifier
Operation

Person
Properties
id
rank
firstName
lastName
address
phone
remarks
status
unitID
functionID
Navigation Properties
LocationResponsibilities
LocationsPersons
Unit
ReceivedMessages
SentMessages
CreatedMessages
Function
ModifiedMessages

Location
Properties
id
fullName
shortName
address
ownerAddress
ownerContact
personResponsibleID
unitID
coordinateID
contact
operationID
from
to
description
Navigation Properties
SlaveConnections
MasterConnections
Coordinate
Operation
Unit
ResponsiblePerson
LocationsPersons
Achievements

Coordinate
Properties
id
altitude
latitudeGPS
longitudeGPS
latitudeLV03
longitudeLV03
latitudeLV95
longitudeLV95
Navigation Properties
Location

Connection
Properties
firstLocationID
secondLocationID
setupPhaseID
teardownPhaseID
Navigation Properties
MasterLocation
SlaveLocation
TeardownPhase
SetupPhase

Function
Properties
id
function
description
Navigation Properties
LocationsPersons
Persons

LocationPerson
Properties
locationID
personID
functionID
Navigation Properties
Location
Person
Function

Phase
Properties
id
name
description
order
operationID
from
to
Navigation Properties
TeardownConnections
SetupConnections
Operation
Achievements

Achievement
Properties
id
description
type
phaseID
locationID
due
achieved
Navigation Properties
Location
Phase

Service
Properties
id
name
description
from
to
Navigation Properties
Operations

Operation
Properties
id
name
type
serviceID
Navigation Properties
Locations
Service
Phases
Messages

AdminRole
Properties
ID
ADGroup
ConfigurationID
Navigation Properties
Configuration

Configuration
Properties
ID
PollInverval
Navigation Properties
AdminRoles
PluginAccesses

PluginAccess
Properties
ID
PluginName
RequireAdmin
ConfigurationID
Navigation Properties
Configuration

Abbildung 29: Datenmodell Location



6.5.3.1 Location

Repräsentiert einen Standort.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identität
FullName	*String*	Der Vollständige Name eines Standortes; Bsp.: „Bronschhofen“
ShortName	*String*	Der Kurzname des Standortes; Bsp.: „BRH“
Description	*Text*	Die Beschreibung des Standortes, mit Ziel, Zweck und Material
Address	*Text*	Die Adresse des Standortes
Contact	*Text*	Die Kontaktinformationen zum Standort; Bsp.: SE-235, SE-240, IMFS-, Fest- oder Mobilnetznummer
OwnerAddress	*Text*	Die Adresse des Landbesitzers
OwnerContact	*Text*	Die Kontaktinformationen des Landbesitzers
From	*DateTime*	Zeitpunkt, zu welchem der Standort bezogen wird
To	*DateTime*	Zeitpunkt, zu welchem der Standort verlassen wird
PersonResponsibleID	*Int*	ID der Person, die für den Standort verantwortlich ist
UnitID	*Int*	ID der Einheit, der der Standort zugeordnet ist
CoordinateID	*Int*	ID der Koordinaten des Standortes
OperationID	*Int*	ID der Operation, der der Standort zugeordnet ist
SlaveConnections	*Relation*	Die Menge der Verbindungen, für welche dieser Standort der übergeordnete Knotenpunkt ist
MasterConnections	*Relation*	Die Menge der Verbindungen, für welche dieser Standort der untergeordnete Knotenpunkt ist
Coordinate	*Relation*	Die Koordinaten des Standortes
Operation	*Relation*	Die Operation, der der Standort zugeordnet ist
Unit	*Relation*	Die Einheit, der der Standort zugeordnet ist
ResponsiblePerson	*Relation*	Die Person, die für den Standort verantwortlich ist
Achievments	*Relation*	Die Menge der Errungenschaften, die von diesem Standort erreicht werden müssen

Tabelle 43 Beschreibung Entität Location



6.5.3.2 LocationPerson

Stellt eine Zuordnungstabelle zwischen Personen und Standorten dar.

Property	Typ	Beschreibung
LocationID	*Int*	Die ID des Standortes, dem die Person zugeteilt ist
PersonID	*Int*	Die ID der Person, die dem Standort zugeordnet ist
FunctionID	*Int*	Die ID der Funktion, die die Person auf dem Standort hat
Person	*Relation*	Der Standort, dem die Person zugeteilt ist
Location	*Relation*	Die Person, die dem Standort zugeordnet ist
Function	*Relation*	Die Funktion, die die Person auf dem Standort hat

Tabelle 44 Beschreibung Entität LocationPerson

6.5.3.3 Coordinate

Kapselt verschiedene Koordinatenformate, die dann zum Beispiel von einem Standort verwendet werden können.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identität
Altitude	*Int*	Höhe über Meer in Meter
LatitudeGPS	*Double*	Breitengrad-Koordinate im GPS Format
LongitudeGPS	*Double*	Längengrad-Koordinate im GPS Format
LatitudeLV03	*Double*	Breitengrad-Koordinate im LV03 Format
LongitudeLV03	*Double*	Längengrad-Koordinate im LV03 Format
LatitudeLV95	*Double*	Breitengrad-Koordinate im LV95 Format
LongitudeLV95	*Double*	Längengrad-Koordinate im LV95 Format
Location	*Relation*	Der Standort, dem die Koordinaten zugeordnet sind

Tabelle 45 Beschreibung Entität Coordinate



6.5.3.4 Connection

Repräsentiert eine Kommunikationsverbindung zwischen zwei Standorten.

Property	Typ	Beschreibung
FirstLocationID	*Int*	Die ID des übergeordneten Standortes
SecondLocationID	*Int*	Die ID des untergeordneten Standortes
SetupPhaseID	*Int*	Die ID der Phase, in welcher die Verbindung hergestellt wird
TeardownPhaseID	*Int*	Die ID der Phase, in welcher die Verbindung abgebaut wird
MasterLocation	*Relation*	Der übergeordnete Standort
SlaveLocation	*Relation*	Der untergeordnete Standort
SetupPhase	*Relation*	Die Phase, in welcher die Verbindung hergestellt wird
TeardownPhase	*Relation*	Die Phase, in welcher die Verbindung abgebaut wird

Tabelle 46 Beschreibung Entität Connection



6.5.4 Person

Unit
Properties
id
name
parentID
Navigation Properties
Locations
Persons

Location
Properties
id
fullName
shortName
address
ownerAddress
ownerContact
personResponsibleID
unitID
coordinateID
contact
operationID
from
to
description
Navigation Properties
SlaveConnections
MasterConnections
Coordinate
Operation
Unit
ResponsiblePerson
LocationsPersons
Achievements

Coordinate
Properties
id
altitude
latitudeGPS
longitudeGPS
latitudeLV03
longitudeLV03
latitudeLV95
longitudeLV95
Navigation Properties
Location

Message
Properties
id
senderID
recipientID
priority
type
text
creatorID
created
modified
modifierID
operationID
isDraft
Navigation Properties
Recipient
Sender
Creator
Modifier
Operation

Person
Properties
id
rank
firstName
lastName
address
phone
remarks
status
unitID
functionID
Navigation Properties
LocationResponsibilities
LocationsPersons
Unit
ReceivedMessages
SentMessages
CreatedMessages
Function
ModifiedMessages

Connection
Properties
firstLocationID
secondLocationID
setupPhaseID
teardownPhaseID
Navigation Properties
MasterLocation
SlaveLocation
TeardownPhase
SetupPhase

Function
Properties
id
function
description
Navigation Properties
LocationsPersons
Persons

LocationPerson
Properties
locationID
personID
functionID
Navigation Properties
Location
Person
Function

Phase
Properties
id
name
description
order
operationID
from
to
Navigation Properties
TeardownConnections
SetupConnections
Operation
Achievements

Achievement
Properties
id
description
type
phaseID
locationID
due
achieved
Navigation Properties
Location
Phase

Service
Properties
id
name
description
from
to
Navigation Properties
Operations

Operation
Properties
id
name
type
serviceID
Navigation Properties
Locations
Service
Phases
Messages

AdminRole
Properties
ID
ADGroup
ConfigurationID
Navigation Properties
Configuration

Configuration
Properties
ID
PollInverval
Navigation Properties
AdminRoles
PluginAccesses

PluginAccess
Properties
ID
PluginName
RequireAdmin
ConfigurationID
Navigation Properties
Configuration

Abbildung 30: Datenmodell Person



6.5.4.1 Person

Repräsentiert eine Person der WK-Einheit.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identifikation
Rank	*Enum*	Der Rang der Person
FirstName	*Sting*	Die Vornamen der Person
LastName	*Sting*	Der Nachname der Person
Address	*Text*	Die Adresse der Person
Phone	*Sting*	Die Telefonnummer der Person
Remarks	*Text*	Zusätzliche Bemerkungen zur Person wie Krankheiten
Status	*Enum*	Der Status der Person
UnitID	*Int*	Die ID der Einheit, der die Person zugeordnet ist
FunctionID	*Int*	Die ID der Funktion, der die Person zugeordnet ist
LocationResponsibilities	*Relation*	Die Menge der Standorte, für die die Person verantwortlich ist
LocationsPersons	*Relation*	Die Menge der Standorte, der die Person zugeordnet ist
Unit	*Relation*	Die Einheit der Person
ReceivedMessages	*Relation*	Die Menge der Nachrichten, die diese Person empfangen hat
SentMessages	*Relation*	Die Menge der Nachrichten, die diese Person gesendet hat
CreatedMessages	*Relation*	Die Menge der Nachrichten, die diese Person erstellt hat, was aber nicht bedeuten muss dass er deren Absender ist
ModifiedMessages	*Relation*	Die Menge der Nachrichten, die diese Person bearbeitet hat
Function	*Relation*	Die Funktion, die die Person einnimmt

Tabelle 47 Beschreibung Entität Person

Status von Personen

Eine Person kann jeweils einem der folgenden Status zugeordnet werden:

- Im Dienst
- Ausser Dienst
- Kurzurlaub
- Wochenendurlaub
- Verwundet
- Krank
- Handlungsunfähig
- Gefangen
- Vermisst



Rang

Eine Person der Schweizer Armee kann einen der folgenden Ränge innehaben: [5]

- Rekrut
- Soldat
- Gefreiter
- Obergefreiter
- Korporal
- Wachtmeister
- Fourier
- Feldweibel
- Stabsadjutant
- Hauptadjundant
- Chefadjundant
- Leutnant
- Oberleutnant
- Hauptmann
- Major
- Oberstleutnant
- Oberst
- Fachof
- Brigadier
- Divisionär
- Korpskommandant
- General

6.5.4.2 Unit

Repräsentiert eine der WK untergeordnete Einheit wie zum Beispiel eine Kompanie, einen Zug oder ein Detachement.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identifikation
Name	*String*	Name der Einheit
ParentID	*Int*	Die ID der der Einheit übergeordneten Einheit
Parent	*Relation*	Die der Einheit übergeordnete Einheit

Tabelle 48 Beschreibung Entität Unit

6.5.4.3 Function

Repräsentiert eine Funktion, die eine Person annehmen kann.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identifikation
Name	*String*	Bezeichung der Funktion; Bsp.: „Stao Chef“
Description	*Text*	Beschreibung der Funktion
Persons	*Relation*	Personen, die diese Funktion innehaben
LocationsPersons	*Relation*	Personen, welche diese Funktion für einen Standort haben

Tabelle 49 Beschreibung Entität Function

[5] [url3]: Dienstgrade der Schweizer Armee, http://de.wikipedia.org/w/index.php?title=Dienstgrade_der_Schweizer_Armee&oldid=103606123 (15.05.2012)



6.5.5 Meldungen

Abbildung 31: Datenmodell Message



6.5.5.1 Message

Repräsentiert eine Meldung. Hauptverwendungsort einer solchen Meldung ist die Triage, die Statusmeldungen von den Standorten entgegennimmt und diese an die Eist weiterleitet. Es ist aber auch möglich, Meldungen irgendeiner Natur zu erfassen und an eine im System vorhandene Person zu senden.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identifikation
SenderID	*Int*	Die ID des Absenders der Nachricht
ReceipientID	*Int*	Die ID des Empfängers der Nachricht
Priority	*Enum*	Die Priorität der Meldung
Type	*Enum*	Der Typ der Meldung
Text	*Text*	Die zusätzlichen Informationen zur Meldung
CreatorID	*Int*	Die ID der Person die die Nachricht erstellt hat
Created	*DateTime*	Der Zeitpunkt zu welchem die Meldung erstellt wurde
ModifierID	*Int*	Die ID der Person die die Meldung zuletzt bearbeitet hat
Modified	*DateTime*	Der Zeitpunkt zu dem die Meldung zuletzt bearbeitet wurde
OperationID	*Int*	Die ID der Operation, in welcher die Meldung erfasst wurde
IsDraft	*Boolean*	Legt fest ob die Meldung nur ein Entwurf ist
Receipient	*Relation*	Der Empfänger der Meldung
Sender	*Relation*	Der Absender der Meldung
Creator	*Relation*	Der Ersteller der Meldung
Modifier	*Relation*	Die Person die die Meldung zuletzt geändert hat
Operation	*Relation*	Die Operation, in welcher die Meldung erfasst wurde

Tabelle 50 Beschreibung Entität Message



Priorität

Die Priorität der Meldung wird im Ermessen des Erstellers gesetzt und ist nicht als Verbindlich zu betrachten. Folgende Prioritäten können einer Meldung zugeordnet werden:

- Normal
- Mittel
- Hoch

Typ

Der Typ der Meldung kann ein direkter Indikator für einen Standort und dessen Errungen darstellen, oder aber eine völlig andere Natur haben. Folgende Type können einer Meldung zugeordnet werden:

- Berreitschaftsraum erreicht
- Bereitschaftsraum verlassen
- Standort erreicht
- Standort verlassen
- Rückruf erforderlich
- Bestellung
- Antrag
- Information



6.5.6 Plugins

Um der Pluginorientierten Struktur den nötigen Daten- und Sicherheitskontext zu schaffen, müssen einige abstrakte Tabellen angelegt werden, die keine Beziehung zu echten Gegebenheiten haben.

Unit
- Properties: id, name, parentID
- Navigation Properties: Locations, Persons

Location
- Properties: id, fullName, shortName, address, ownerAddress, ownerContact, personResponsibleID, unitID, coordinateID, contact, operationID, from, to, description
- Navigation Properties: SlaveConnections, MasterConnections, Coordinate, Operation, Unit, ResponsiblePerson, LocationsPersons, Achievements

Coordinate
- Properties: id, altitude, latitudeGPS, longitudeGPS, latitudeLV03, longitudeLV03, latitudeLV95, longitudeLV95
- Navigation Properties: Location

Message
- Properties: id, senderID, recipientID, priority, type, text, creatorID, created, modified, modifierID, operationID, isDraft
- Navigation Properties: Recipient, Sender, Creator, Modifier, Operation

Person
- Properties: id, rank, firstName, lastName, address, phone, remarks, status, unitID, functionID
- Navigation Properties: LocationResponsibilities, LocationsPersons, Unit, ReceivedMessages, SentMessages, CreatedMessages, Function, ModifiedMessages

Connection
- Properties: firstLocationID, secondLocationID, setupPhaseID, teardownPhaseID
- Navigation Properties: MasterLocation, SlaveLocation, TeardownPhase, SetupPhase

Function
- Properties: id, function, description
- Navigation Properties: LocationsPersons, Persons

LocationPerson
- Properties: locationID, personID, functionID
- Navigation Properties: Location, Person, Function

Phase
- Properties: id, name, description, order, operationID, from, to
- Navigation Properties: TeardownConnections, SetupConnections, Operation, Achievements

Achievement
- Properties: id, description, type, phaseID, locationID, due, achieved
- Navigation Properties: Location, Phase

Service
- Properties: id, name, description, from, to
- Navigation Properties: Operations

Operation
- Properties: id, name, type, serviceID
- Navigation Properties: Locations, Service, Phases, Messages

AdminRole
- Properties: ID, ADGroup, ConfigurationID
- Navigation Properties: Configuration

Configuration
- Properties: ID, PollInverval
- Navigation Properties: AdminRoles, PluginAccesses

PluginAccess
- Properties: ID, PluginName, RequireAdmin, ConfigurationID
- Navigation Properties: Configuration

Abbildung 32: Datenmodell Plugin



6.5.6.1 AdminRole

Diese Tabelle legt fest, welche im ActiveDirectory des Teplas vorhandenen Rollen im EistCockpit Administrationszugriff haben. Administratoren können Dienstleistungen erstellen.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identifikation
ADGroup	*String*	Der Name der Gruppe im ActiveDirectory, die innerhalb des EistCockpits als Administratoren gehandhabt werden soll
ConfigurationID	*Int*	Die ID der der AdminRole zugeordneten Konfiguration
Configuration	*Relation*	Die der AdminRole zugeordnete Konfiguration

Tabelle 51 Beschreibung Entität AdminRole

6.5.6.2 Configuration

Diese Tabelle beinhaltet alle für das EistCockpit notwendigen Konfigurationsfelder.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identifikation
PollIntervall	*Int*	Sagt aus, wie oft die Clientapplikation die Datenbank abfragen soll
AdminRoles	*Relation*	Die AdminRoles, die der Konfiguration zugeordnet sind
PluginAccesses	*Relation*	Die Plugins, die der Konfiguration zugeordnet sind

Tabelle 52 Beschreibung Entität Configuration

6.5.6.3 PluginAccess

Diese Tabelle beinhaltet Alle ladbaren Plugins für eine Konfiguration und gibt an, ob ein Benutzer über Administratorenrechte verfügen muss, um das Plugin zu laden.

Property	Typ	Beschreibung
ID	*Int*	Eindeutige Identifikation
PluginName	*String*	Name des Plugins
RequireAdmin	*Boolean*	Sagt aus, ob Administrationsrechte benötigt werden, um das Plugin zu laden
ConfigurationID	*Int*	Die ID der dem Plugin zugeordneten Konfiguration
Configuration	*Relation*	Die dem Plugin zugeordnete Konfiguration

Tabelle 53 Beschreibung Entität PluginAccess



6.6 User Interface

6.6.1 Kognitiver Test

6.6.1.1 Einleitung

Der Kognitive Test wurde in einem ersten Schritt im Team durchgeführt. Dabei wurde ein erster Entwurf mit dem Balsamiq Mockup Tool erstellt. Anschliessend wurde er zusammen mit den anderen Team-Kameraden besprochen. Jeder überlegte sich, wo allfällige Schwierigkeiten auftauchen könnten und was man eventuell verbessern müsste.

6.6.1.2 Beteiligte Personen

- Diego Steiner
- David Schöttl
- Remo Waltenspül

6.6.1.3 Mockups

Host Application

Die Host-Application ist das Grundgerüst für die Anwendung und zeigt in einem Container Fenster die Inhalte der Plugins an.

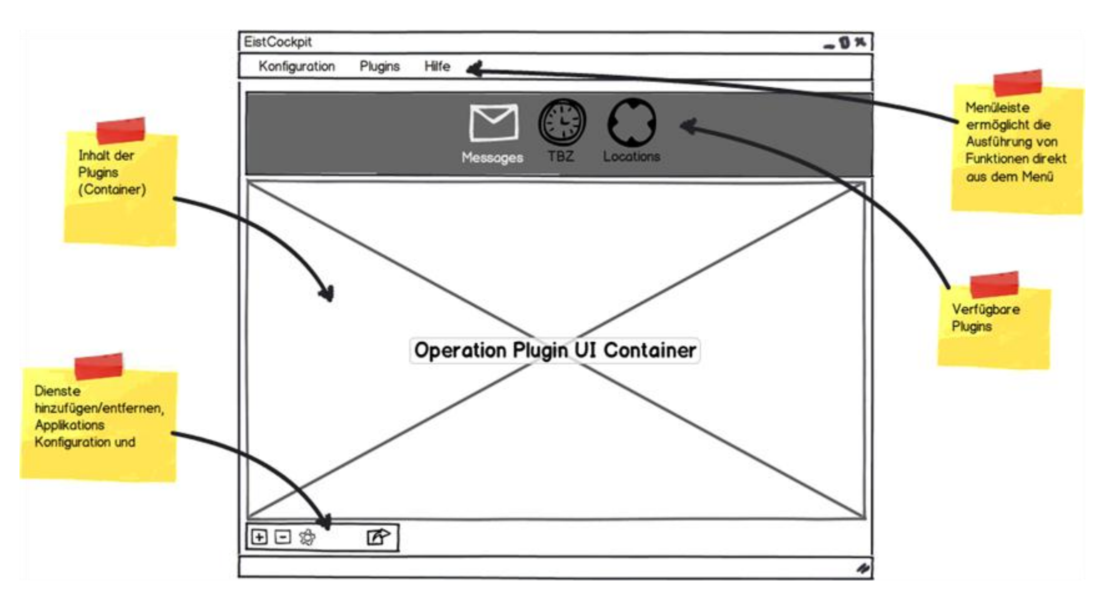


Abbildung 33: Host Application Mockup



Plugin Meldung - Meldungen erfassen (Triage)

In diesem Tab des Plugins Meldungen können neue Meldungen erfasst werden.

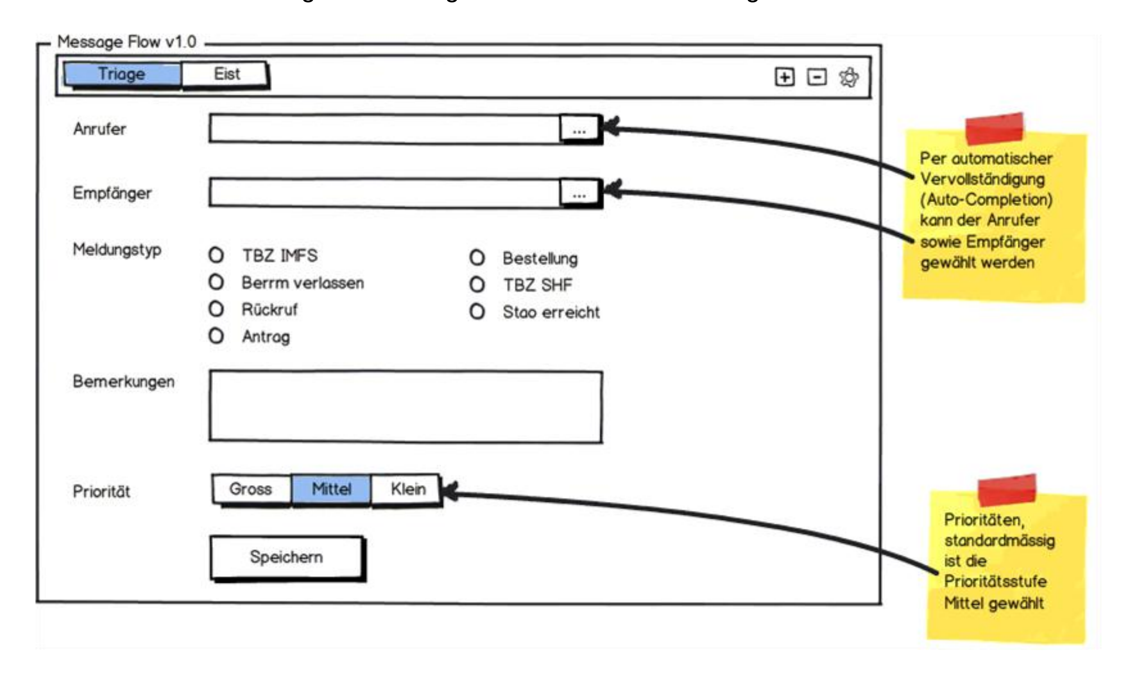


Abbildung 34: Create Message Mockup

Plugin Meldung - Übersicht Meldungen (Eist)

Die Übersicht dient vorwiegend der Eist zum Anzeigen aktueller Meldungen.

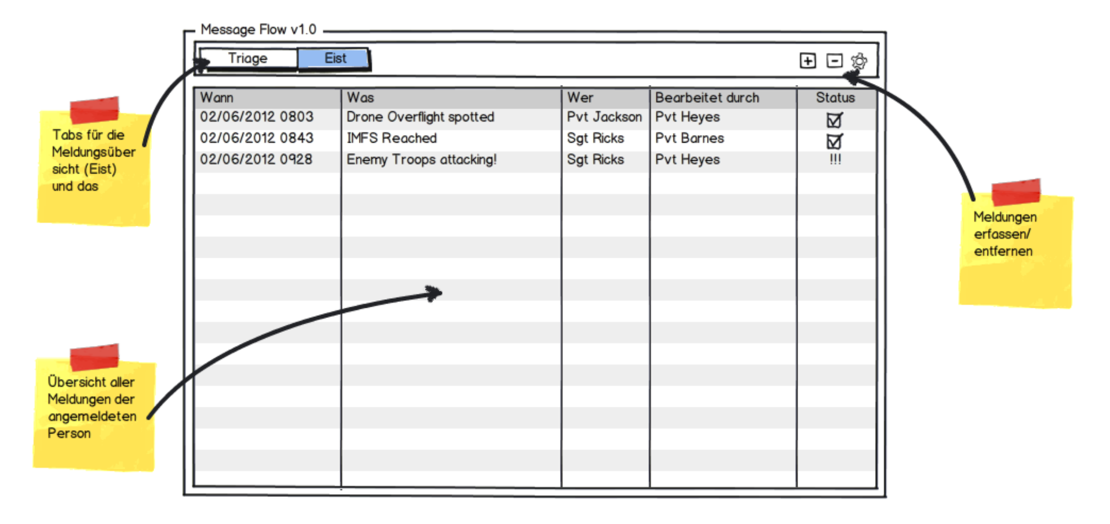


Abbildung 35: Message Overview Mockup



Plugin Standorte

Das Plugin Standort dient dazu neue Standorte zu erfassen sowie bisherige Standorte zu bearbeiten. Ausserdem erlaubt es die bisher erfassten Standorte zu einer bestimmten Übung anzuzeigen. Die nachfolgende Abbildung Verweis Abbildung 36: Location Mockup gibt einen Überblick über den ersten Entwurf.

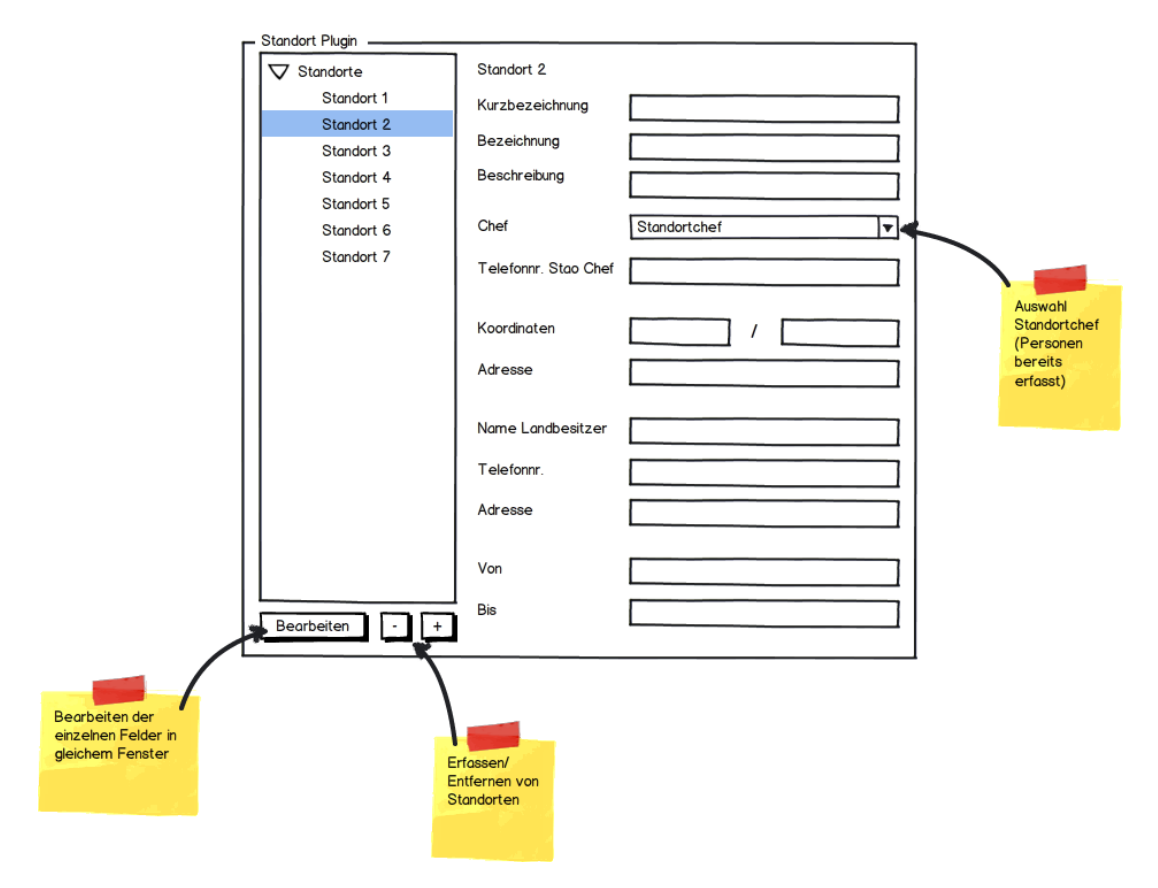


Abbildung 36: Location Mockup



6.6.1.4 Erkenntnisse (Findings)

Während dem kognitiven Test innerhalb des Teams wurden folgende Feststellungen gemacht. Anhand dieser Erkenntnisse wurden die Mockups überarbeitet.

Plugin Standorte

- Die Schaltflächen unterhalb der Auflistung von Standorten sind nicht sehr intuitiv angeordnet.
- Auf den ersten Blick könnte der Benutzer aufgrund der deaktivierten Eingabefelder irritiert sein.
- Die Bezeichnungen heben sich nicht vom Inhalt ab, dadurch fällt es schwer den Text von den Beschriftungen zu unterscheiden.
- Das Datum muss von Hand eingegeben werden, da würde sich ein DateTimePicker anbieten.
- Die Beschreibung sowie die Adresse sollten auf mehreren Zeilen eingegeben werden können, dies ist mit den derzeitigen Eingabefeldern nicht möglich.

Plugin Meldung - Triage

- Meldungstypen sind schlecht gruppiert, durch eine Neuanordnung kann Benutzbarkeit verbessert werden.
- Beim Erfassen von Meldungen werden keine Hilfestellungen in Form von Text dargestellt
- Bei der Eingabe des Anrufers wird keine automatische Vervollständigung angeboten
- Die Schaltflächen mit den drei Prioritäten enthalten nicht die typischen Symbole, welche bei der Meldungsübersicht verwendet werden.
- Die Reihenfolge der Meldungsattribute in der Übersicht ist nicht sehr sinnvoll gesetzt.

Plugin Meldung - Eist

- Es existiert keine Suchleiste um einzelne Meldungen zu suchen
- Es fehlt an erweiterten Suchmöglichkeiten (Filter), um nach spezifischen Kriterien zu filtern
- Meldungen können nicht nach Standort gruppiert werden.

Allgemein (Host Application)

- Das Register Eist zeigt die Meldungsübersicht und sollte deshalb vor dem Register Triage vorkommen.
- Menüleiste nicht mehr zeitgemäss, besser aussagekräftige Schaltflächen und angemessene Kurztastenbefehle
- Der Kontext bzw. die konkrete Übung eines bestimmten Militärdienstes kann nicht ausgewählt werden.



6.6.1.5 Mockups nach Überarbeitung

Host Application

Die Host Application wurde um eine Kontextleiste mit allen verfügbaren Übungen erweitert. Diese Auswahl gilt anschliessend für sämtliche Plugins, welche auf die gewählte Übung spezifisch reagieren können.

Die Menüleiste wurde ganz weggelassen, da man im Team der Meinung ist, dass durch eine intuitive Benutzeroberfläche eine Menüleiste überflüssig wird.

Weiter wurden die Schaltflächen für das Hinzufügen/Entfernen von Übungen sowie Konfigurationseinstellungen in den Kontextbereich (Host-Plugin) verschoben.

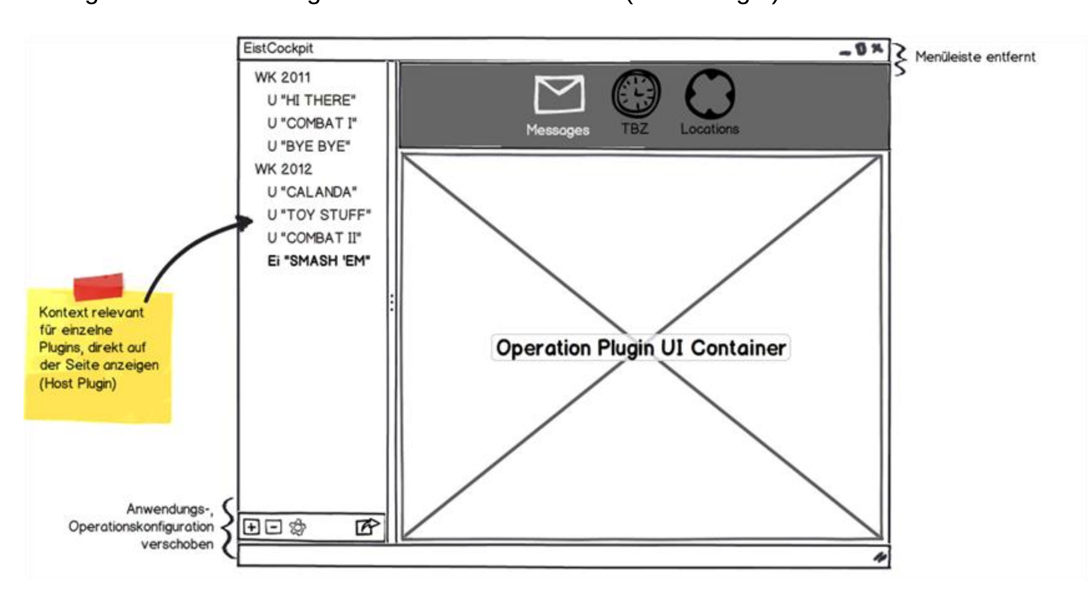


Abbildung 37: Host Application Service Mockup



Meldungen erfassen (Triage)

Die grösste Änderung betreffend dem Formular zum Erfassen von Meldungen wurde bei der Benutzerunterstützung anhand von Hilfetext vorgenommen.

Die neue Version wurde weiter durch eine automatische Vervollständigung beim Erfassen von Namen ergänzt, ausserdem wurden passende Symbole bei den Prioritätsschaltflächen hinzugefügt.

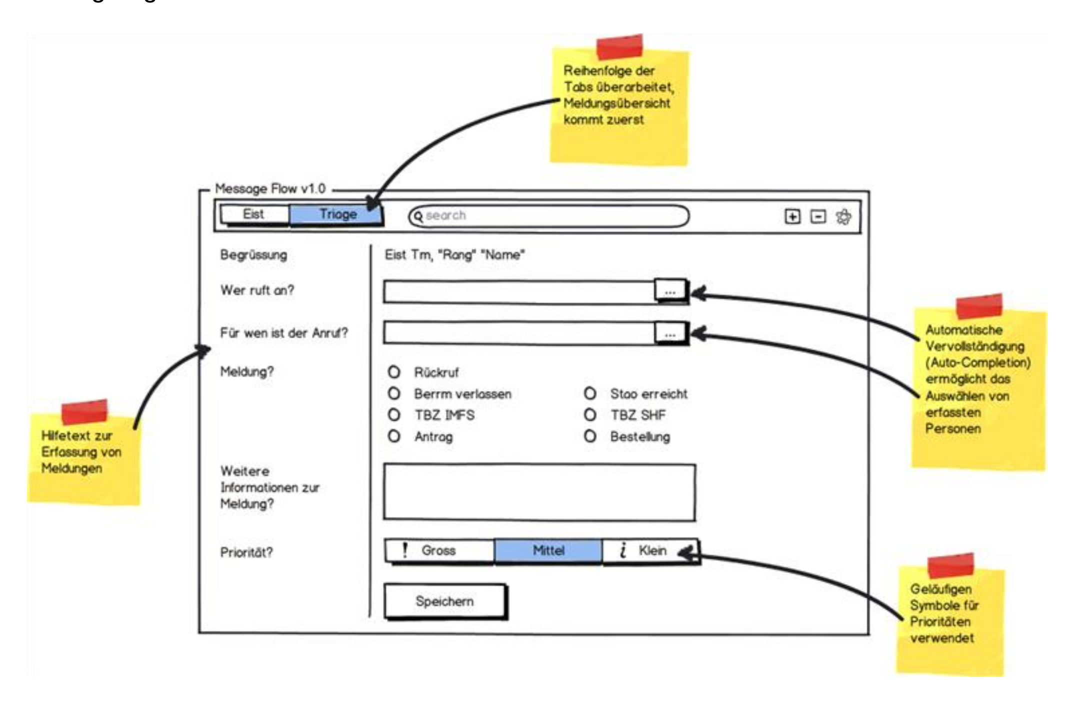


Abbildung 38: Create Message Mockup



Übersicht Meldungen (Eist)

Im Vordergrund der Meldungsübersicht stand sicherlich die bessere Übersichtlichkeit sowie Such-, Filtermöglichkeiten um die Meldungen nach bestimmten Kriterien auszuwählen. Dazu wurde die Übersicht um eine Suchleiste und einen erweiterten Filter erweitert.

Der zweite Punkt, welcher für mehr Klarheit sorgen soll, ist das Einrücken bzw. Gruppieren von Meldungen anhand des Standorts.

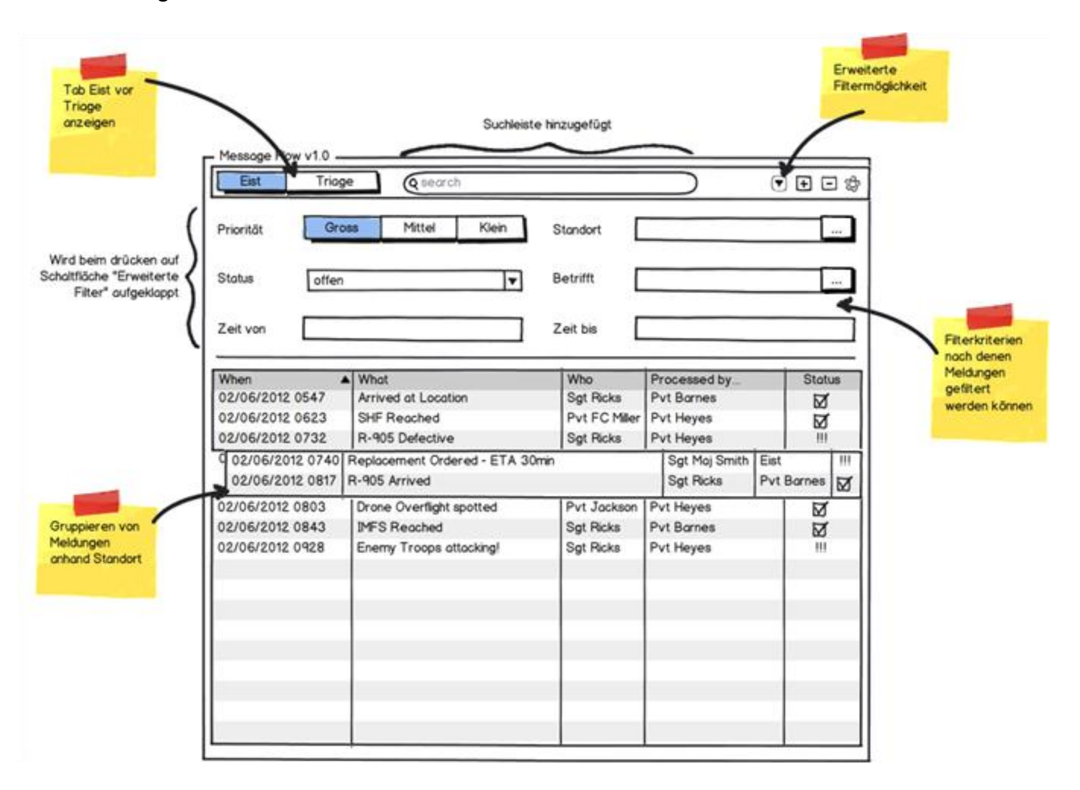


Abbildung 39: Message Filter Mockup



Plugin Standorte

Die auffälligste Neuerung nach dem kognitiven Test ist sicherlich, dass die Bearbeitung eines Standorts in einem modalen Fenster ausgeführt wird. Dies hauptsächlich aufgrund der einfacheren Kontrolle, da im Falle einer direkten Bearbeitung, die Auswahl eines anderen Kontexts (Übung) gesperrt werden müsste. Da im Plugin Container nur Standortinformationen angezeigt werden, werden normale Textfelder anstelle von deaktivierten Eingabefelder verwendet. Weiter werden für eine bessere Benutzbarkeit geeignetere User Controls wie zum Beispiel ein DateTimePicker benutzt.

Abbildung 40: Location Mockup Redesign



6.6.2 Heuristischer Test

6.6.2.1 Meldung erfassen

1	Sichtbarkeit des System-Status	Operation-Kontext jederzeit sichtbar
2	Enger Bezug zwischen System und realer Welt	Stimmt mit Meldezettel überein
3	Nutzerkontrolle und Freiheit	
4	Konsistenz & Konformität mit Standards	
5	Fehler-Vorbeugung	Completion & Validation erfüllen diese Kriterien
6	Besser Sichtbarkeit als Sicherinnern-Müssen	Meldung erfassen Button muss angeschrieben sein
7	Flexibilität und Nutzungseffizienz	
8	Ästhetik und minimalistischer Aufbau	
9	Nutzern helfen, Fehler zu bemerken, zu diagnostizieren und zu beheben	
10	Hilfe und Dokumentation	Hilfetext könnte in separatem Fenster angezeigt werden

Tabelle 54 Heuristischer Test Meldung erfassen



6.6.2.2 Meldungsübersicht

1	Sichtbarkeit des System-Status	
2	Enger Bezug zwischen System und realer Welt	Spaltenüberschriften müssen konsistent sein
3	Nutzerkontrolle und Freiheit	
4	Konsistenz & Konformität mit Standards	
5	Fehler-Vorbeugung	
6	Besser Sichtbarkeit als Sicherinnern-Müssen	
7	Flexibilität und Nutzungseffizienz	
8	Ästhetik und minimalistischer Aufbau	
9	Nutzern helfen, Fehler zu bemerken, zu diagnostizieren und zu beheben	
10	Hilfe und Dokumentation	

Tabelle 55 Heuristischer Test Meldungsübersicht

6.6.2.3 Plugin Standorte

1	Sichtbarkeit des System-Status	Statusinformation in Statusleiste
2	Enger Bezug zwischen System und realer Welt	
3	Nutzerkontrolle und Freiheit	
4	Konsistenz & Konformität mit Standards	
5	Fehler-Vorbeugung	Validierung der Eingaben
6	Besser Sichtbarkeit als Sicherinnern-Müssen	
7	Flexibilität und Nutzungseffizienz	Datum & Zeit per DateTimePicker
8	Ästhetik und minimalistischer Aufbau	
9	Nutzern helfen, Fehler zu bemerken, zu diagnostizieren und zu beheben	
10	Hilfe und Dokumentation	Hilfestellungen zu erwartetem Eingabeformat könnte angezeigt werden.

Tabelle 56 Heuristischer Test Plugin Standorte



6.6.3 Papiertprototyp Test

6.6.3.1 Einführung zu den Tests

Wir möchten Sie herzlich willkommen heissen zu unserem Papierprototyp Test. Nachfolgend haben wir zwei Szenarien vorbereitet, dabei bitten wir Sie die Aufträge laut vorzulesen, bevor Sie mit den Aufgaben beginnen. Zudem wären wir froh, wenn Sie uns Ihre Gedanken während dem Test laufend mitteilen könnten, indem Sie einen inneren Monolog führen.

6.6.3.2 Testszenarien

Auftrag 1 Meldung erfassen

Sie arbeiten während Ihrer Dienstleistungspflicht in der Triage und sind verantwortlich für die Entgegennahme von Telefonanrufen externer Standorte. Zurzeit ist es 15:00 Uhr, es herrscht keine grosse Hektik, da nicht so viele Telefone eingehen. In diesem Wiederholungskurs wird das erste Mal die neue Anwendung Eist Cockpit verwendet, welche die ehemaligen Meldezettel vollumfänglich ersetzen soll.

Plötzlich beginnt das Telefon zu klingeln, Sie nehmen den Hörer auf und sprechen mit dem Anrufer (gespielt von rwaltens), zeitgleich erfassen Sie die Meldungen mit sämtlichen notwendigen Informationen.

Folgender Teil muss nicht mehr vorgelesen werden (Exakter Ablauf)

Sie erhalten eine Meldung von einem gewissen Vermittlungsbetreuer (Vm-Betr.) Sdt Waltenspül, welcher berichtet, dass die Telematikbereitschafszeit (TBZ) erreicht wurde. Schon während dem Gespräch wählen Sie den Namen des Anrufers sowie die Option TBZ erreicht aus. Da bekannt ist, dass sämtliche Telematik Belange in den Zuständigkeitsbereich des FGG3 gehört, wählen Sie als Empfänger die Gruppe FGG3. Bevor Sie die Meldung übermitteln können, müssen Sie noch die Priorität setzen, wobei es sich bei einer TBZ Meldung um eine normale Priorität handelt. Eine Beschreibung ist für diese Meldung nicht erforderlich, deshalb senden Sie die Meldung direkt an den definierten Empfänger (FGG3).

Auftrag 2 Meldungen nach bestimmtem Kriterium filtern

Sie arbeiten während Ihrer Dienstleistungspflicht in der Eist Tm und sind verantwortlich für die Kontrolle der Einhaltung der Telematikbereitschaftszeiten externer Standorte. Der Chef der Eist, Hptm Müller, betritt den Raum und verlangt umgänglich eine Übersicht über die eingegangenen TBZ Meldungen der letzten Stunde. Sie filtern die Ansicht nach den genannten Kriterien und zeigen Diese dem Hptm.



6.6.3.3 Testauswertungen

Sdt Daniel Günthensberger (Mitarbeiter in der Triage sowie Eist)

Durchgeführt am Mi 18.04.12 17:00 Uhr

Allgemein

- Die Hilfe Schaltfläche ist unter den verfügbaren Plugins positioniert, jedoch handelt es sich nicht um ein Plugin. Die Hilfe sollte von jedem Plugin eigenständig implementiert werden.
- Mit Hörer kann fast nicht getippt werden

Szenario 1: Meldung erfassen

- Tab Name "Triage" & "Eist Tm" sind auf den ersten Blick nicht ganz klar
- Empfänger ist bei TBZ Meldung untergegangen (aber soweit i.O. weil das Feld dann automatisch ausgefüllt wird). Platzierung überdenken
- Statt auf Senden wurde auf die Schaltfläche Speichern gedrückt

Szenario 2: Meldungen nach bestimmtem Kriterium filtern

- Filter Button nicht gefunden
- Es war nicht ganz klar, wo man Meldungen anzeigen kann (Testperson drückte zuerst auf Plugin-Schaltfläche TBZ)
- Filter müssen mit Feldnamen übereinstimmen

Gfr. Thomas Corbat (In der Rolle als Triage Mitarbeiter)

Durchgeführt am Mo 30.04.12 16:00 Uhr

Allgemein

Suchfeld beibehalten, ev. mit Erweitert anschreiben, dass man das Filtersymbol nicht suchen muss

Szenario 1: Meldung erfassen

- Speichern/Senden Buttons müssen Aussagekräftigere Namen haben
- Die Meldungstypen-Optionen sollten noch spezifischer sein
- Die Meldungstypen-Optionen sollten nach Recently-Used angeordnet sein
- Rückruf / Zur Info unterscheiden
- Sortierung bei Auswahlliste nach Standortbezeichnung nicht nach Standortnummer

Szenario 2 Meldungen nach bestimmtem Kriterium filtern

- Formularfelder mit der Zeit mit aktueller Uhrzeit vorausfüllen
- Spaltenüberschriften vereinheitlichen
- Konsistenz prüfen (Prioritäten)
- Datetimepicker für die Auswahl der Uhrzeit



6.6.3.4 Erkenntnisse aus Paper-Prototype Test

Der Papier-Prototyp Test war äusserst nützlich, da dadurch viele neue Erkenntnisse gesammelt werden konnten. Die beiden Testkandidaten haben ihre Aufgabe sehr seriös durchgeführt, und dabei viele Verbesserungsvorschläge sowie kleinere Hinweise gegeben um das System möglichst benutzerfreundlich zu gestalten.

Viele Aussagen haben sich zudem gedeckt, was bedeutet die beiden Tester haben zum Teil ähnliche Mängel festgestellt.

Für eine zukünftige Version der Benutzeroberfläche wird primär das Augenmerk auf die Konsistenz (Beschriftungen, Schaltflächen etc.), klare aussagekräftige Beschreibungen sowie eine übersichtliche Darstellung (Wichtige Schaltflächen hervorheben) gerichtet.

6.6.3.5 Eingescannter Prototyp V1

Übersicht Meldungen

Diese Ansicht wird primär in der Einsatzstelle Telematik (Eist Tm) verwendet. Es ermöglicht dem Benutzer die Anzeige aller aktuellen Meldungen (siehe Abbildung 41: Paper Prototype Message Overview).



Abbildung 41: Paper Prototype Message Overview



Formular Meldungen erfassen

Dieses Formular dient dazu, Meldungen in wenigen Schritten zu erfassen. Dabei wird unterschieden zwischen dem Senden einer Meldung und dem Zwischenspeichern. Beim Zwischenspeichern werden die Meldungen in einer "Sandbox" aufbewahrt und können so später fertig ausgefüllt werden (siehe Abbildung 42: Paper Prototype Create Message).

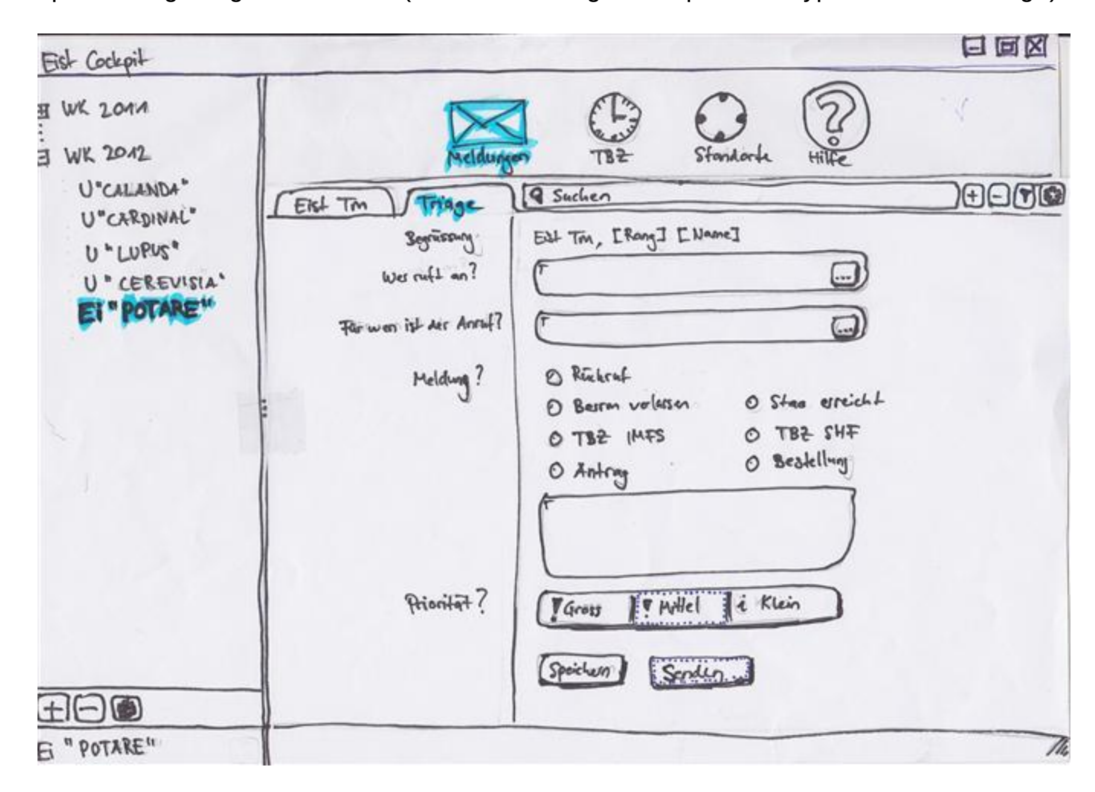


Abbildung 42: Paper Prototype Create Message



Meldung erfasst

Die folgende Abbildung macht die Benutzung der Statusleiste ersichtlich. Sobald eine Meldung erfasst wurde, wird der Benutzer über die Statusleiste informiert (siehe Abbildung 43: Paper Prototype Message Created).

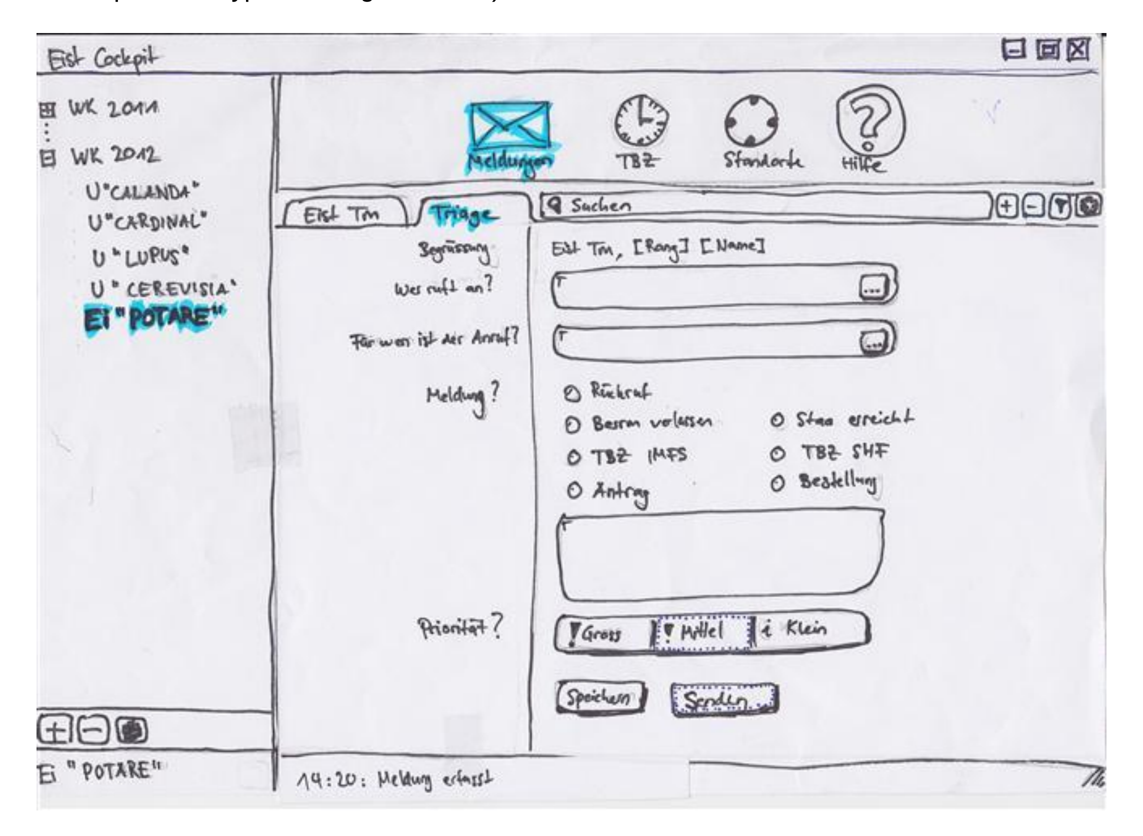


Abbildung 43: Paper Prototype Message Created



Filter

Mittels des Filters können die Meldungen nach bestimmten Kriterien ausgewählt werden. Die Filteransicht wird erst sichtbar, wenn auf das kleine Filtersymbol gedrückt wird (siehe Abbildung 44: Paper Prototype Message Overview).

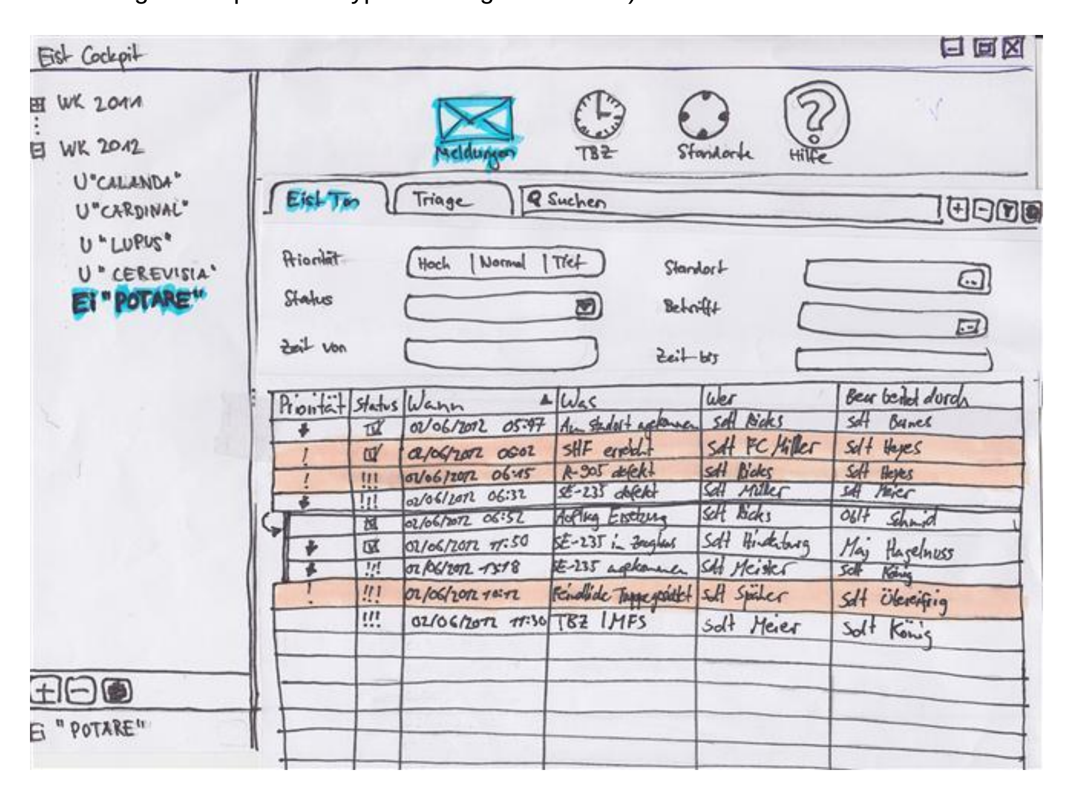


Abbildung 44: Paper Prototype Message Overview



Gefilterte Meldungen

Die untenstehende Abbildung zeigt die Situation nach einer Filterung nach einem bestimmten Begriff (z.B. IMFS).

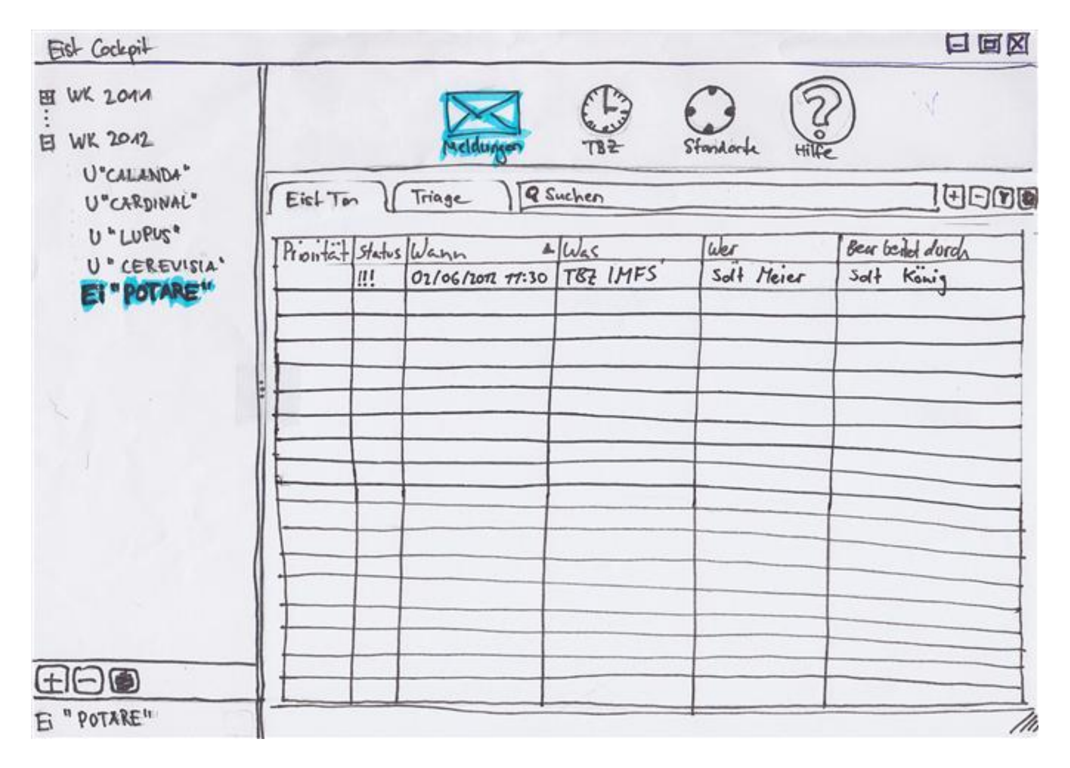


Abbildung 45: Paper Prototype filtered Messages



6.6.4 Redesign

Anhand der Erkenntnisse aus den durchgeführten Papier-Prototypen Tests wurde das gesamte Design nochmals überarbeitet. Besonderes Augenmerk wurde auf eine einfachere Benutzbarkeit sowie einer intuitiveren Bedienung gerichtet.

6.6.4.1 Host Applikation

Redesign Entscheide

Vergrössern der Schaltfläche zur Konfiguration von Dienstleistungen (Note 1)

Die Schaltflächen waren für einen weniger geübten Benutzer zu klein.

Überschrift in Kontextfenster (Note 2)

Gibt dem Benutzer eine bessere Übersicht

Hilfe aus Plugin-leiste entfernt (Note 3)

Bei der Schaltfläche Hilfe handelt es sich nicht um ein Plugin, deshalb sollte es nicht direkt in der Plugin-leiste sondern spezifisch im Plugin UI Container angezeigt werden.

Mockup nach Redesign

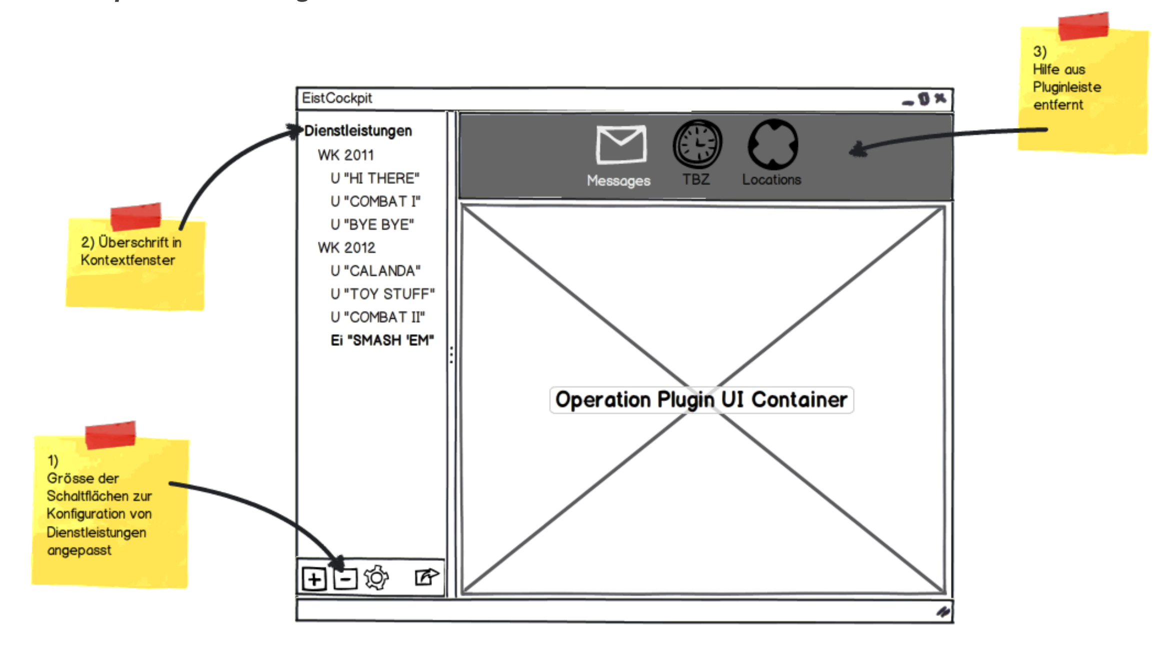


Abbildung 46: Host Application Service Mockup Redesign



6.6.4.2 Meldungsübersicht

Redesign Entscheide

Aktuelle Zeit bei der erweiterten Suche (Note 1)

Häufig werden die vergangenen Meldungen bis zum derzeitigen Zeitpunkt angezeigt. Aus diesem Grund wird standardmässig die gegenwärtige Zeit im Feld „Zeit bis“ eingeblendet.

Kontextauswahl hinzugefügt (Note 2)

Da mit steigender Zahl von Meldungen die Übersicht schnell verloren gehen kann, wird auf eine weitere Kontextübersicht gesetzt. Mit dieser zusätzlichen Ansicht können alle Meldungen eines bestimmten Standorts oder einer Phase ausgewählt werden.

Tab Namen Triage & Eist wurden ersetzt durch Übersicht & Entwürfe (Note 3)

Die Bezeichnungen Triage & Eist waren zu wenig aussagekräftig, stattdessen werden die beiden Begriffe Übersicht und Entwürfe benutzt. Wobei das Tab Entwürfe alle Meldungen enthält, welche noch nicht versandt wurden.

Konsistente Prioritätsbezeichnungen (Note 4)

Die Prioritäten sollten einheitlich gewählt werden, damit keine Verwirrung entsteht.

Alle Beschriftungen rechtsbündig ausgerichtet (Note 5)

Diese Darstellung sieht moderner aus und führt zu klaren Linien was zu einer grösseren Ästhetik der Oberfläche führt.

Schaltfläche für Erweiterte Suchfunktionalität um Text ergänzt (Note 6)

Aus dem Papier Prototyp Test ging eindeutig hervor, dass die erweiterten Suchmöglichkeiten nur schwer zu finden waren. Um dies zu vermeiden wird die Schaltfläche um einen zusätzlichen beschreibenden Text („Erweitert“) ergänzt.

Schaltflächen gruppiert (Note 7)

Die verschiedenen Funktionen werden so gruppiert, dass das Auffinden und lokalisieren einfacher ist.

Hilfe Schaltfläche hinzugefügt (Note 8)

Das Plugin Nachrichten wird um eine Schaltfläche „Hilfe“ erweitert. Diese Schaltfläche wurde von der Pluginleiste in den Pluginbereich verschoben, da es sich bei der Hilfe um eine spezifische Hilfestellung zum Plugin handelt.

Datetimepicker für die Auswahl der Uhrzeit (Note 9)

Für eine einfachere und intuitivere Bedienung wird für die Eingabe der Zeit ein Datetimepicker Control verwendet.

Eindeutige Beschriftungen gewählt für die Tabellenüberschriften sowie die verschiedenen Filterkriterien (Note 10)

Damit die Konsistenz gewährleistet ist und der Benutzer nicht irritiert wird, werden sämtliche Tabellenüberschriften gleich benannt wie die Filterkriterien.



Mockup nach Redesign

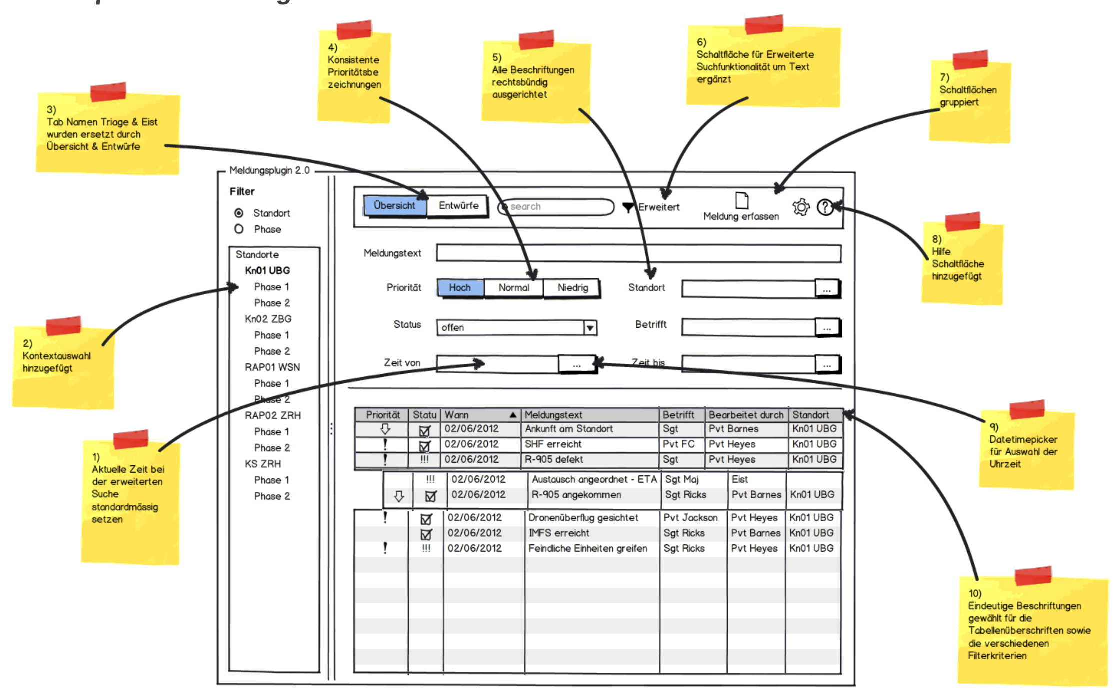


Abbildung 47: Messages Overview Mockup Redesign



6.6.4.3 Meldungen erfassen

Redesign Entscheide

Integriertes Fenster wurde durch Modales Fenster ersetzt (Note 1)

Beim Erfassen der Meldungen werden weder die Kontextleiste noch die Suchleiste etc. benötigt. Es handelt sich weitgehend um ein unabhängiges Fenster, daher wurde das Formular in ein eigenes Modales Fenster ausgelagert.

Zusätzlicher Hilfetext rechtsbündig ausgerichtet (Note 2)

Diese Darstellung sieht moderner aus und führt zu klaren Linien was zu einer grösseren Ästhetik der Oberfläche führt.

Sortierung nach Standortbezeichnung nicht nach Standortnummer (Note 3)

Die Standortnummer sagt zu wenig aus, folge dessen wird die Nummer durch die Standortbezeichnung ersetzt.

Weitere Meldungsoption „Zur Information“ (Note 4)

Aus dem Papier-Prototypen Test wurde ersichtlich, dass eine zusätzliche Meldungsoption „Zur Information“ hilfreich wäre, wenn auf eine Meldung keine Antwort mehr benötigt wird.

Gleichen Symbole wie auf Übersicht verwendet (Note 5)

Für ein einheitliches Design wird darauf geachtet, dass für gleiche Funktionen die gleichen Symbole benutzt werden.

Einheitliche Prioritätsbezeichnungen (Note 6)

Neu werden die Prioritäten wie folgt bezeichnet: Hoch, Normal, Niedrig. Dies fördert die Konsistenz und folge dessen die Benutzbarkeit

Beschriftung der Schaltflächen überarbeitet (Note 7)

Die Schaltflächen waren vor dem Redesign zu wenig aussagekräftig. Die Benutzer wurden durch die Namensgebung verunsichert, deshalb wird nun die Schaltfläche zum Übermitteln von Meldungen als „Senden“ und die Schaltfläche zum Zwischenspeichern als „Zwischenspeichern“ beschriftet.

Schaltfläche Senden hervorheben mit grösserer Schrift (Note 8)

Da im Normalfall die Meldung nach dem Erfassen direkt übermittelt wird, wird die Schrift der „Senden“ Schaltfläche grösser dargestellt.

Mockup nach Redesign



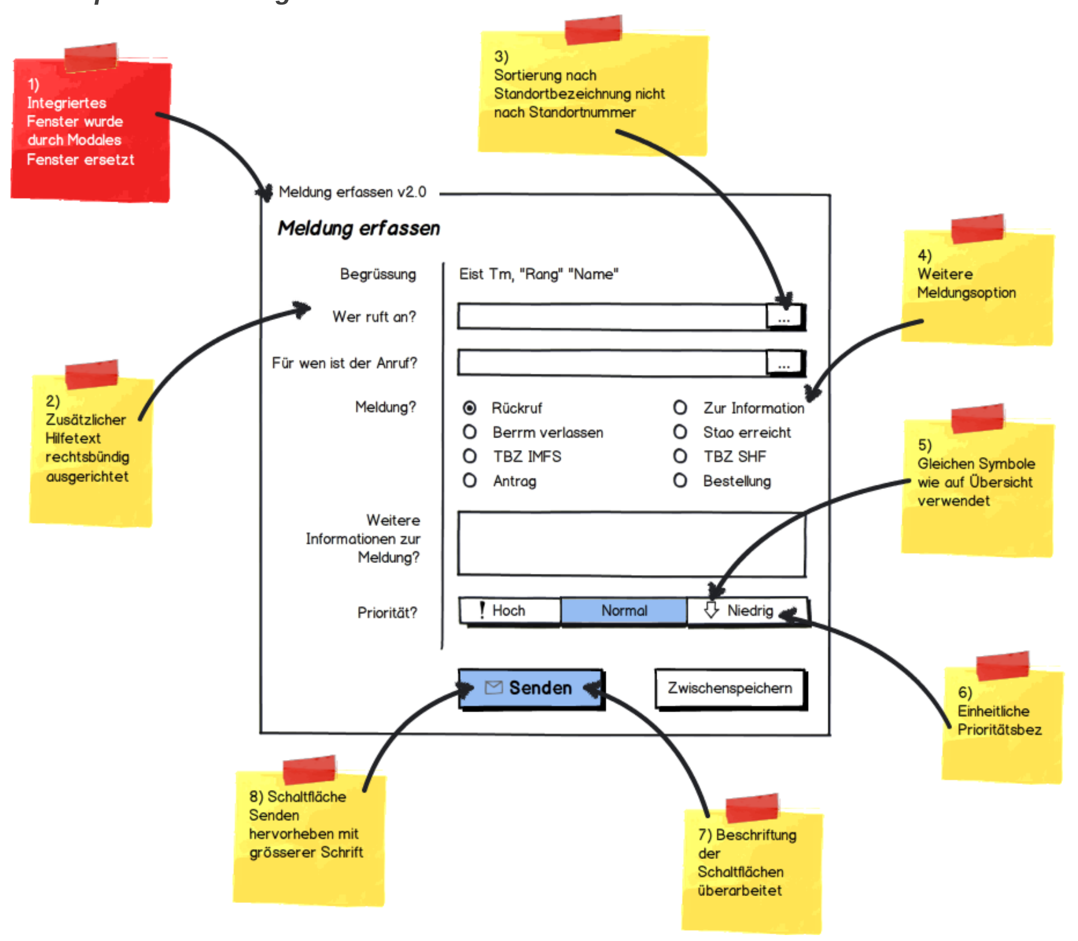


Abbildung 48: Message Mockup Redesign



6.6.4.4 Plugin Standorte

Redesign Entscheide

Treeview wurde ersetzt durch Liste von Standorte (Note 1)

Die Verwendung des Treeview User Controls macht an dieser Stelle keinen Sinn, da mit nur einer dynamischen Hierarchieebene gearbeitet wird. Aus diesem Grund wird es durch eine Listbox ersetzt.

Kombobox für die Auswahl der Einheit (Note 2)

Während der Implementationsphase erkannte man, dass laut dem Datenmodel eine Verbindung zur Einheit bestehen muss. Bis zu diesem Zeitpunkt war es jedoch nicht möglich eine Einheit zu einem Standort auszuwählen.

Eingabemaske für Koordinaten verwendet (Note 3)

Da das Eingabeformat für die Koordinaten sich nie verändert, und stets aus je sechs Zahlen besteht, wird ein spezielles User Control mit einer Eingabemaske verwendet. Damit wird der Benutzer unterstützt die Koordinaten im richtigen Format anzugeben.

Dialog beim Abbrechen sofern Daten verändert wurden. (Note 4)

Falls der Benutzer einen Standort bearbeitet hat und aus Versehen auf die Schaltfläche Abbrechen drückt, sollte eine Meldung erscheinen, die den Benutzer informiert.

Zusätzliche Schaltfläche Abbrechen hinzugefügt (Note 5)

Wenn der Benutzer die Bearbeitung eines Standorts abbrechen möchte, musste er dies bis anhin über den Close Button machen. Es erscheint jedoch intuitiver, wenn dem Benutzer die Möglichkeit gegeben wird über eine spezielle Schaltfläche dies durchzuführen.

Feld für Hinweis zu Validierungsfehlern hinzugefügt (Note 6)

Um dem Benutzer den Grund für das Fehlschlagen der Validierung aufzuzeigen, wird ein Informationsfeld benutzt. Damit wird für den Benutzer klarer ersichtlich warum es nicht möglich ist einen Standort zu erfassen.



Mockup nach Redesign

Standort Plugin

Standorte

Standort 1
Standort 2
Standort 3
Standort 4
Standort 5
Standort 6

\+ - Bearbeiten

Standort 2

Allgemein

Kurzbezeichnung Kurzbez. Standort 2
Bezeichnung Bezeichnung Stao 2
Beschreibung Beschreibung Stao 2
Einheit Mustereinheit
Chef Stao Chef
Telefonnr. Stao Chef 044 938 38 32

Lage

Koordinate 382938 / 193829
Adresse Musterstr. 12

Landbesitzer

Name Landbesitzer Max Muster
Telefonnr. Teststr. 83
Adresse Eine Beschreibung

Belegt

Von 10.05.2011 16:16
Bis 12.05.2011 18:30

1) Treeview wurde ersetzt durch Liste von Standorte

Standort bearbeiten/erfassen

Standort 2

Allgemein

Kurzbezeichnung
Bezeichnung
Beschreibung
Einheit
Chef
Telefonnr. Stao Chef

Lage

Koordinaten ___/_____
Adresse

Landbesitzer

Name Landbesitzer
Telefonnr.
Adresse

Belegt

Von / /
Bis / /

Speichern Abbreche

2) Kombobox für die Auswahl der Einheit

3) Eingabemaske für Koordinaten verwendet

4) Dialog beim Abbrechen sofern Daten verändert wurden

5) Zusätzliche Schaltfläche Abbrechen hinzugefügt

6) Feld für Hinweis zu validierungsfehlern hinzugefügt

Abbildung 49: Location Mockup Redesign



6.6.5 UI Guidelines

Weil das System in einer homogenen Windowsumgebung laufen muss, ist das Design des Userinterfaces an den „Windows User Experience Interaction Guidelines“ [6]von Microsoft Orientiert:

[6] [url4]: Windows User Experience Interaction Guidelines, http://msdn.microsoft.com/de-ch/library/windows/desktop/aa511258.aspx (12.05.2012)



EistCockpit

7 Entwurf

David Schöttl, Remo Waltenspül & Diego Steiner



7.1 Dokumenteninformation

Datum	Version	Änderung	Autor
06.06.2012	1.0	Erste Version des Dokuments	dschöttl
07.06.2012	1.1	Transaktionsmanagement hinzugefügt	dschöttl
08.06.2012	1.2	Review	rwaltens



7.2 Design Entscheide

7.2.1 Systemlandschaft

EistCockpit wird auf der existierenden Systemlandschaft eines Ristl Bat (Richtrahl Bataillon) eingesetzt. Dieses System heisst TEPLAS (**Te**lematik **Pla**nungs **S**ystem) und besteht aus mehreren, z.T. virtualisierten, Servern (Windows Server 2003) sowie einer Anzahl mobiler Arbeitsstationen (Windows XP).

Die Arbeitsstationen sind über LAN mit den Servern verbunden, wobei das Netzwerk komplett isoliert ist – d.h. es besteht keine Verbindung zu anderen Netzwerken (z.B. Internet, etc.). Auf den Arbeitsstationen ist nur ein Basis-Betriebssystem installiert (Windows XP); für die Arbeit meldet sich ein Benutzer via Citrix beim Server an und arbeitet somit innerhalb seiner Session effektiv auf dem Server – die Arbeitsstation ist lediglich ein glorifiziertes Terminal.

Die Benutzerverwaltung geschieht – wie bei Windows Server Umgebungen üblich – mittels AD[7] (Active Directory) wobei die Benutzer gewissen Gruppen zugewiesen sind, welche deren Rechte definieren.[8]

7.2.2 Technische Auflagen

Von Seiten der FU Br 41 (Führungsunterstützungs Brigade 41) wurden bei der Initiierung des Projekts folgende technische Auflagen[9] gemacht:

- Einsetzbarkeit auf den verfügbaren IT-Mittel der Ristl Bat (Teplas oder KP LAN); d.h. es dürfen keine weiteren Systeme oder Geräte in die bestehende Infrastruktur integriert werden.
- Abhängigkeiten bei allfälligen Softwareupdates (Betriebssystemseitig) sind vollumfänglich zu dokumentieren.

7.2.3 Framework & Programmiersprache

Wie im vorherigen Abschnitt dargelegt wurde, wird das System in einer Microsoft Windows Umgebung eingesetzt. Dabei darf zwar keine weitere *Hardware* der Umgebung hinzugefügt werden, wohl jedoch *Software* – sofern dies klar dokumentiert wird. Dies implizert, dass die von EistCockpit benötigte Laufzeitumgebung sowie deren Frameworks installiert werden.

Nachfolgend werden die Varianten für die verwendete Laufzeitumgebung dargelegt.

7.2.3.1 Varianten

7.2.3.2 Variante 1: Java

Java[10] ist eine weit verbreitete und moderne, objektorientierte Programmiersprache, welche besondere Stärken im Bereich 'Netzwerk' zu verzeichnen hat. Des Weiteren bietet Java diverse Frameworks für die server- wie auch clientseitige Entwicklung an.

---

[7] [url5]: Active Directory, http://en.wikipedia.org/wiki/Active_Directory (28.05.2012)
[8] Beispiele für solche Gruppen werden hier aus Gründen der Geheimhaltung nicht genannt.
[9] Zitate aus eMail vom 28.11.2011 zwischen Hptm Michael Klötzli und Gfr Thomas Corbat.



Java ist zudem plattformübergreifend, d.h. eine Java Applikation kann auf jedem Betriebssystem ausgeführt werden – dies jedoch unter der Voraussetzung, dass eine Java Runtime installiert ist.

7.2.3.3 Variante 2: C# & .NET 4.0

Die objektorientierte Programmiersprache C#[11] wird standardmässig für die professionelle Entwicklung unter Windows verwendet. Gleiches gilt für das .NET Framework[12], welches umfangreiche Unterstützung in jeglichen Belangen bietet.

C# wurde später als Java entwickelt, z.T. unter Anlehnung an Java (z.B. Syntax). Die Sprache wurde fortwährend weiter entwickelt und ist mittlerweile um einiges mächtiger als Java.

Applikationen, welche unter der Verwendung des .NET Frameworks entwickelt wurden, können nur auf Windows-Systemen ausgeführt werden; dabei muss die entsprechende .NET Version auf dem System installiert sein.

7.2.3.4 Entscheid

Variante 2 – Für die Entwicklung von EistCockpit wird verwendet:

- Programmiersprache: **C#**
- Framework: **.NET 4.0**

Der Entscheid wird wie folgt begründet:

- Dies Systemumgebung basiert auf Microsoft Windows.
- Ein .NET Framework (Version 2.0) ist bereits installiert; das .NET Framework 4.0 wird von den vorhandenen Betriebssystemen unterstützt.[13]
- Das .NET Framework 4.0 bietet mächtige Technologien wie WCF[14][15] (Windows Communication Foundation) und WPF[16] (Windows Presentation Foundation), welche auf modernen Entwicklungskonzepten, wie z.B. MVVM[17] (Model-View-ViewModel), aufbauen.
- Alle Team-Mitglieder sind erfahren in der .NET-Entwicklung unter C# – zum einen durch Modulbesuche, als auch durch Arbeitserfahrung inkl. Microsoft Zertifizierungen.

---

[10] [url6]: Java (programming language), http://en.wikipedia.org/wiki/Java_(programming_language) (29.05.2012)

[11] [url7]: C Sharp (programming language), http://en.wikipedia.org/wiki/C_Sharp_(programming_language) (29.05.2012)

[12] [url8]: .NET Framework, http://en.wikipedia.org/wiki/.NET_Framework (03.06.2012)

[13] [url9]: Download Microsoft .NET Framework 4.0 (eigenständiger Installer), http://www.microsoft.com/de-de/download/details.aspx?id=17718 (03.06.2012)

[14] [url10]: Windows Communication Foundation, http://en.wikipedia.org/wiki/Windows_Communication_Foundation (04.06.2012)

[15] [url11]: What is Windows Communication Foundation, http://msdn.microsoft.com/en-us/library/ms731082.aspx (04.06.2012)

[16] [url12]: Windows Presentation Foundation, http://en.wikipedia.org/wiki/Windows_Presentation_Foundation (04.06.2012)

[17] [url13]: Model View ViewModel, http://en.wikipedia.org/wiki/MVVM (04.06.2012)



7.2.4 Server-Client System

Aufgrund der vorhandenen Hardware, der Verfügbarkeit eines Netzwerks, sowie der inhärenten Natur des Systems liegt eine Server-Client-Architektur, basierend auf .NET, auf der Hand.

Während serverseitig in jedem Fall ein Service als Schnittstelle zum Einsatz kommt, tritt clientseitig die Frage auf, welche Form der Client haben soll. Nachfolgend werden Varianten für die Client-Software näher beleuchtet.

7.2.4.1 Varianten

7.2.4.2 Variante 1: Webapplikation

In der heutigen Zeit von Web 2.0 drängt sich der Gedanke an eine Webapplikation schnell in den Vordergrund, sobald die Begriffe 'Service' und 'Netzwerk' fallen.

Unter dem Begriff 'Webapplikation' wird hierbei eine Applikation verstanden, mit welcher innerhalb eines Browsers, z.B. Internet Explorer, interagiert wird.

Vorteile:

- Minimaler bis nicht-existenter Installationsaufwand, da nur ein Web Browser zur Verfügung stehen muss.
- Kein Aufwand durch Software Updates.

Nachteile:

- User Interface schwierig umzusetzen.
- Probleme mit unterschiedlichen Browser Versionen und Implementationen.
- Erfahrungsgemäss sehr aufwändig in der Entwicklung.
- Einschränkungen, welche aus der "Laufzeitumgebung" 'Web Browser' resultieren.

7.2.4.3 Variante 2: Rich Client

Unter dem Begriff 'Rich Client' wird hierbei eine Applikation verstanden, welche effektiv vom Client-Betriebssystem ausgeführt wird – mit eigenständiger Executable Binary, Fenstern, Dialogen und dergleichen.

Unter der Verwendung von WCF für die Netzwerk-Kommunikation sowie WPF für die Benutzeroberfläche lassen sich morderne, benutzerfreundliche und gut wartbare Applikationen in angemessener Zeit umsetzen.

Vorteile:

- Spezifisches User Interface Design
- Höhere Performance
- Breite Einsatzmöglichkeiten, da Systemzugriff möglich.
- WCF & WPF sind sehr mächtig
- Keine Abhängigkeit von Web Browsern – in einer Systemumgebung, welche per Definition keinen Browser benötigt.

Nachteile:

- Installationsaufwand bei Updates
- .NET Framwork 4.0 muss installiert werden



7.2.4.4 Variante 3: Mischform Webapplikation/Rich Client

Als weitere Variante wurde eine Mischform zwischen Webapplikation und Rich Client erdacht:

- Ein Teil von EistCockpit, z.B. die Nachrichten-Erfassung in der Triage, wird als Webapplikation entwickelt.
- Andere Teile von EistCockpit werden in Form eines Rich Clients entwickelt, welcher z.B. in der Netzleitung der Eist zum Einsatz kommt.

Der Gedanke zu dieser Variante gründet in der Überlegung, dass z.B. ein Triage-Mitarbeiter im Range eines Soldaten nicht den gleichen Funktions-Umfang benötigt, wie ein Feldweibel während der Leitung des Richtstrahl-Netzwerks.

Diese Variante bringt jedoch einen erhöhten Aufwand mit sich, da verschiedene Arten von User Interfaces auf verschiedenen Plattformen gepflegt werden müssen.

7.2.4.5 Entscheid

Variante 2 – Der clientseitige Teil von EistCockpit wird als **Rich Client** umgesetzt.

Der Entscheid wird wie folgt begründet:

- Insbesondere in Bezug auf das User Interface – ein sehr wichtiger Aspekt dieses Systems – bietet ein WPF-basierter Rich Client umfangreichere Möglichkeiten bei kleinerem Aufwand, im Vergleich zu einer Webapplikation.
- Da die Systemumgebung nicht mit dem Internet verbunden ist und auch sonst keinen Web Browser benötigt, wirkt die Voraussetzung eines aktuellen Web Browsers absurd.
- Der Einsatz des .NET Framework 4.0 hat keinen Einfluss auf bereits bestehenden Applikationen.
- Einschränkungen im Funktionsumfang können mittels Zugriffsbeschränkungen und dergleichen umgesetzt werden, was eine Mischform unnötig macht.

Die Verwendung von Rich Clients im Zusammenhang mit Webservices wird auch im IEEE-Beitrag [grechanik07] "Composing Integrated Systems Using GUI-Based Applications And Web Services" diskutiert.



7.2.5 Service

Wie bereits erwähnt kommt für die Kommunikation zwischen Service und Client die WCF Komponente des .NET Framework 4.0 zum Einsatz. WCF bietet die Möglichkeit, verschiedene Typen von Webservices zu entwickeln.

Nachfolgend werden Varianten für Webservices beschrieben. Aufgrund einer Nutzwert- und Sensitivitätsanalyse wird sodann der Entscheid für einen Typ von Webservice begründet.

7.2.5.1 Varianten

7.2.5.2 Variante 1: WSDL/SOAP

WCF verwendet standardmässig WSDL[18] (Web Services Description Language) in Verbindung mit SOAP[19] (Simple Object Access Protocol) für die Bereitstellung eines Webservice und die Kommunikation damit.

Aufgrund der Metadaten zum Webservice, welche durch WSDL beschrieben werden, kann automatisch eine Clientimplementation des Webservice generiert werden. Das XML-basierte SOAP bietet mitunter, durch XML Schemas definierte, Validierungsmöglichkeiten für den Nachrichtenaustausch. In einer Umgebung, wo ausschliesslich .NET Technologie zum Einsatz kommt, funktioniert WSDL/SOAP praktisch transparent.

Probleme tauchen erfahrungsgemäss dann auf, wenn mit anderer Technologie als .NET darauf zugegriffen wird (z.B. Mobile Clients auf Basis von iOS); diese Probleme haben oft ihren Ursprung in den Schema-Definitionen.[20]

Der Einsatz von XML bei SOAP bringt Vorteile. Gleichzeitig hat dies aber zum Nachteil, dass die Grösse einzelner Nachrichten, verursacht durch XML Markup und Redundanzen in den Schemas, in einem schlechten Verhältnis zum Informationsgehalt steht.

7.2.5.3 Variante 2: REST/JSON

WCF unterstützt – bei entsprechender Konfiguration – auch Webservices, welche über eine REST[21] (Representational State Transfer) Schnittstelle angesprochen werden können und Daten im JSON[22] (JavaScript Object Notation) Format liefern.

Im Gegensatz zu WSDL stellt REST den Datenzugriff über spezielle, virtuelle URLs zur Verfügung und verwendet für die grundlegenden CRUD Operationen entsprechende HTTP Kommandos. Damit kann sogar mit einem Web Browser auf die rohen Daten zugegriffen werden.

JSON stellt ein Datenformat dar, welches seinen Ursprung in der Programmiersprache JavaScript hat. Es ist ein sehr simples Format, welches – einfach ausgedrückt – Strukturen, bestehend aus Arrays und Dictionaries, beschreibt.

---

[18] [url14]: Web Services Description Language, http://en.wikipedia.org/wiki/WSDL (04.06.2012)
[19] [url15]: SOAP, http://en.wikipedia.org/wiki/SOAP (04.06.2012)
[20] Ein Sprichwort des Autors zum Thema 'SOAP', welches eine Ausgeburt des SE-2 Projekts 'BudgetGuard' ist, lautet: "Mit Seife muss man vorsichtig sein – sonst rutscht man aus!"
[21] [url16]: Representational state transfer, http://en.wikipedia.org/wiki/REST (04.06.2012)
[22] [url17]JSON, http://en.wikipedia.org/wiki/JSON (04.06.2012)



Im Gegensatz zu SOAP bietet JSON keine ausgefeilten Möglichkeiten für die Validierung von Nachrichten. Dafür ist das Datenformat extrem kompakt [2], woraus ein sehr gutes Verhältnis zwischen Grösse und Informationsgehalt für einzelne Nachrichten resultiert. Zudem ist JSON für einen Menschen leichter lesbar als SOAP, was beim Debugging hilfreich sein kann.

7.2.5.4 Nutzwertanalyse

Nutzwertanalyse: Service Typ						
			Variante 1 WSDL/SOAP		Variante 2 REST/JSON	
#	Kriterium	Gewichtung [1..5]	Bewertung [1..5]	Total	Bewertung [1..5]	Total
1	Programmieraufwand	2	5	10	4	8
2	Kompatibilität	5	3	15	5	25
3	Nachrichtengrösse	3	1	3	5	15
4	Lesbarkeit	4	2	8	4	16
	Total Punkte			36		64
	Rang			2		1

Anmerkung: Höhere Gewichtungen/Bewertungen sind wichtiger/besser.

Tabelle 57 Nutzertanalyse

7.2.5.5 Sesitivitätsanalyse

In der Sensitivitätsanalyse wird untersucht, wie stark sich eine Änderung auf das Gesamtergebnis der Nutzwertanalyse auswirken würde.

Die Nutzwertanalyse gewichtet die Kriterien 'Kompatibilität' und 'Lesbarkeit' sehr stark. Der Grund dafür ist, dass ein möglichst offenes und einfaches Format verwendet werden soll – dies auch hinsichtlich eines angedachten iOS Clients für die Zukunft.

Während Variante 2 'REST/JSON' durchwegs hohe Bewertungen für die einzelnen Kriterien erhält, wird Variante 1 'WSDL/SOAP' eher mittelmässig bis schlecht bewertet – besonders in den zwei stärker gewichteten Kriterien.

Variante 1 müsste daher eine signifikante Steigerung in der Bewertung von drei der vier Kriterien erhalten, damit die Rangierung beeinflusst wird; dies ist auch klar aufgrund der Total Punkte für die Varianten ersichtlich.

7.2.5.6 Entscheid

Variante 2 – Es wird ein Webservice unter Verwendung von **REST/JSON** implementiert.

Der Entscheid wird durch das Ergebnis der Nutzwertanalyse, unter Berücksichtigung der Sensitivitätsanalyse, begründet.

7.2.6 Plugins

Mitunter ein Ziel von EistCockpit ist, dass die Software einfach mit weiteren Features ausgebaut werden kann; dabei soll der Installations- bzw. Updateaufwand minimal gehalten werden.

Aus diesem Grund verwendet EistCockpit eine Plugin-Architektur, welche unter dem Einsatz von MEF (Microsoft Extensibility Framework) implementiert ist. Für eine detaillierte



Beschreibung des Lademechanismus der Plugins wird auf das Dokument "Prototypen", Abschnitt "MEF-Plugin", verwiesen.

Bei der Konzeption von EistCockpit wurde erkannt, dass, abhängig vom Einsatzzweck, verschiedene Typen von Plugins benötigt werden. Dies gründet in der Entscheidung, dass praktisch *alle* Teile von EistCockpit als Plugin implementiert werden: Die EistCockpit Host Applikation bietet lediglich ein Grundgerüst; die effektiven User Interfaces der verschiedenen Features werden aus den Plugins geladen und im Grundgerüst eingebettet.

Weiter wurde erkannt, dass nicht alle Benutzer den gleichen Umfang an Features benötigen. So sollte z.B. ein Triage Mitarbeiter im Range eines Soldaten keinen Zugriff auf das Service Manager Plugin erhalten – denn die Erfassung von Dienstleistungen und Operationen fällt in den Kompetenzbereich des Kaders. Realisiert wird diese Zugriffskontrolle mittels Mapping der AD Benutzergruppe auf die zwei EistCockpit Benutzergruppen 'User' bzw. 'Administrator': Plugins, welche Administratoren-Rechte voraussetzen, werden nur angezeigt, wenn der Benutzer als 'Administrator' erkannt wird.[23]

Die verschiedenen Typen von Plugins werden nachfolgend im Detail beschrieben.

7.2.6.1 UI Containers

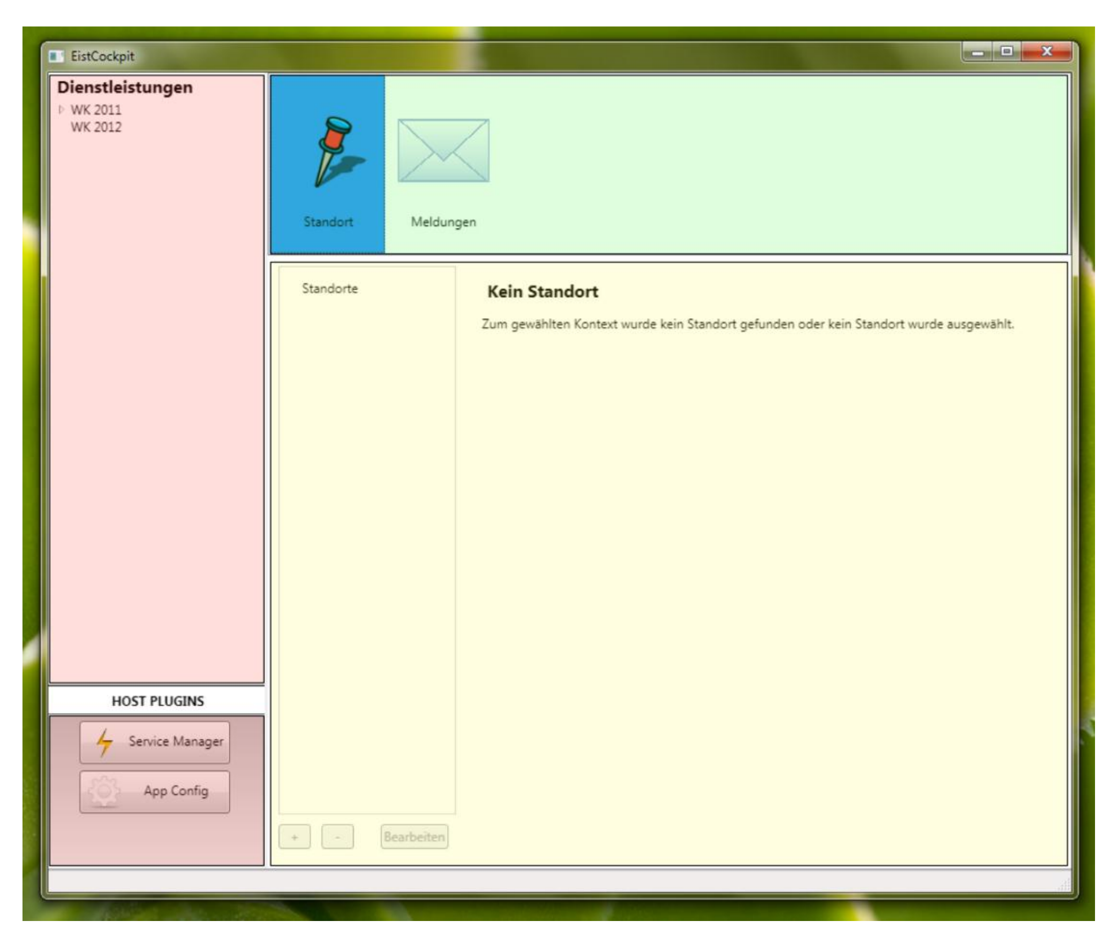


Abbildung 50 UI Container

Die Abbildung zeigt das Main Window von EistCockpit, wobei die einzelnen Bereiche, welche Plugin-spezifische Inhalte darstellen, farblich markiert wurden.

---

[23] Die Grundlagen für diese Zugriffskontrolle sind im Rahmen der Semesterarbeit bereits umgesetzt worden; das Konzept wurde im Sinne eines Proof-of-Concept mit Erfolg ausprobiert. Effektiv aktiviert wird dieses Feature aber erst im Rahmen der Bachelorarbeit.



7.2.6.2 Host Plugins

Host Plugins sind Plugins, welche spezielle und grundlegende Features auf Applikations-Ebene implementieren. Im Allgemeinen verwenden solche Plugins ein eigenständiges Fester/Dialog für ihr UI.

In der obigen Abbildung "UI Containers" sind zwei rot eingefärbte Bereiche auszumachen; der obere Bereich enthält das UI für das Services Plugin – ein Host Plugin, welches als Ausnahme *kein* eigenes Fenster verwendet. Der untere Bereich listet Buttons für die anderen Host-Plugins auf: Service Manager (zur Verwaltung von Dienstleistungen und Operationen), sowie App Config (für die Applikations-Konfiguration).

7.2.6.3 Operation Plugins

Operation Plugins implementieren Features, welche aufgrund ihrer Natur an einen Operations-Kontext gebunden sind. Solche Plugins stellen ihr UI innerhalb des, in der obigen Abbildung "UI Containers" gelb eingefärbten, Bereichs dar.

Der grüne Bereich in der erwähnten Abbildung listet die installierten Operation Plugins auf und ermöglicht den Zugriff darauf.

7.2.6.4 Standalone Plugins

Standalone Plugins implementieren Features, welche nicht zwingend etwas mit dem Einsatzzweck von EistCockpit zu tun haben. Dieser Typ von Plugin verwendet, ähnlich wie Host Plugins, ein eigenständiges Fenster für sein UI.

Einige Beispiele/Ideen für Standalone Plugins:

- Wiki
- Vereins-Verwaltung
- Pausenkiosk-Verwaltung

Standalone Plugins sind erst konzeptionell angedacht; weder im Rahmen der Semesterarbeit noch im Rahmen der Bachelorarbeit ist die Implementation eines Standalone Plugins vorgesehen.

Der Zugriff auf bzw. die Auflistung von Standalone Plugins dürfte ähnlich wie bei den Host Plugins gehandhabt werden; dies wurde aber noch nicht genauer ausgearbeitet.

7.2.6.5 Plugin Implementation

Zur Unterscheidung der verschiedenen Plugin-Typen wurde eine Hierarchie aus Interfaces definiert, wobei ein Plugin einges Typs das entsprechende Interface implementieren muss.

Folgende Interfaces wurden definiert:

- **IPlugin** – Diese Interface stellt das Basis Interface dar; Interfaces für die einzelnen Typen erben von IPlugin und erweitern es entsprechend. IPlugin kommt direkt nie zum Einsatz.
- **IHostPlugin** – Dieses Interface definiert die Schnittstelle für Host Plugins.
- **IAppConfigPlugin** – Da es sich beim App Config Plugin um ein einmaliges und sehr spezielles Plugin handelt, wurde dieses Interface, welches IHostPlugin erweitert, eingeführt. Es wird ausschliesslich vom App Config Plugin verwendet und definiert zusätzliche Properties, welche für die Verwaltung der Plugins benötigt werden.
- **IOperationPlugin** – Dieses Interface definiert die Schnittstelle für Operation Plugins.
- **IStandalonePlugin** – Dieses Interface definiert die Schnittstelle für Standalone Plugins.



7.2.6.6 Struktur

Prinzipiell wird jedes Plugin in einer eigenen Solution entwickelt, welche jeweils über folgende Subprojekte verfügt:

- **Loader** – Das Loader-Projekt implementiert das entsprechende Plugin Interface.
- **ViewModels** – Das ViewModel-Projekt implementiert die View Models, welche vom Plugin verwendet werden.
- **Views** – Das Views-Projekt implementiert das User Interface für das Plugin.

7.2.6.7 Vererbungshierarchie Plugin Interfaces

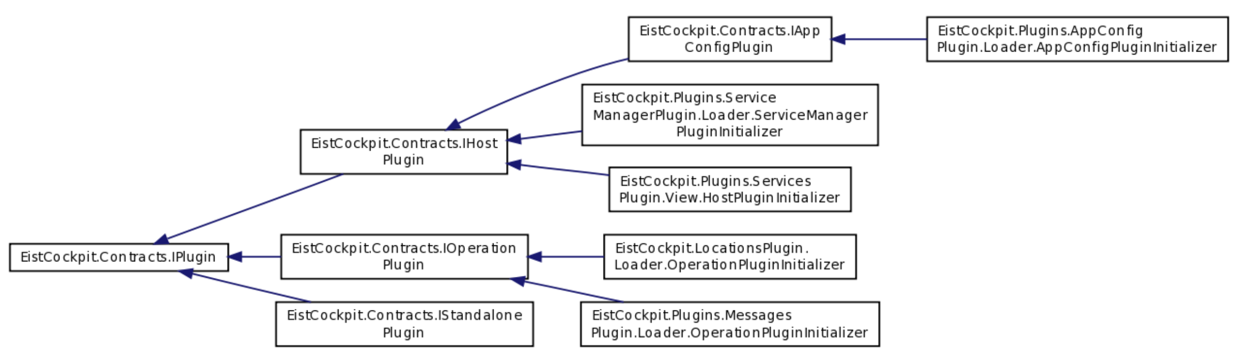


Abbildung 51 Vererbungshierarchie Plugin Interfaces

Die obige Abbildung zeigt die Vererbungshierarchie der Plugin Interfaces sowie die implementierenden Plugins.

7.2.7 Transaktions Management

In verteilten Systemen, welche auf eine gemeinsame Datenbank zugreifen, tritt immer die Problematik auf, wie konkurrenzierende Updates für einen Datensatz gehandhabt werden sollen.

Unter dem Einsatz von WSDL/SOAP Webservices wird hierfür nach dem sog. Optimistic Concurrency Control Prinzip vorgegangen[24]. Bei der Optimistic Concurrency Control wird davon ausgegangen, dass Konflikte zwischen Transaktions-Operationen eher selten auftreten; dies hat positive Auswirkungen auf die Performanz des Datenzugriffs, da Locks und dergleichen entfallen. Bei einem Konflikt wird dies dem Client mitgeteilt, damit dieser entsprechend reagieren kann.

Bei der Verwendung von REST/JSON Webservices – wie im vorliegenden Fall für EistCockpit – ist die Unterstützung für Optimistic Concurrency Control schwieriger umzusetzen; trotzdem wäre ein solcher Mechanismus wünschenswert. Der ACM-Beitrag [hiane09] "An Optimistic Technique for Transaction Control using REST Architectural Style" untersucht diese Thematik und zeigt eine Lösung auf, wie Optimistic Concurreny Control für einen REST/JSON umgesetzt werden kann.

Im Rahmen der Studienarbeit wird aus Gründen des Zeitmangels auf ein entsprechendes Transaktionsmanagement verzichtet; dies ist jedoch für die Bachelorarbeit vorgesehen.

---

[24] Siehe auch Unterlagen zum HSR Modul 'Microsoft Technologien – MsTe'.



7.3 Architektur

7.3.1 Systemübersicht

Version 1.0, 2012-06-05

ARCHITECTURE OVERVIEW

EistCockpit
with Major Components

NETWORK (LAN)

EISTCOCKPIT RICH CLIENT

- Client Application
- .NET-based, WPF
- Used by Eist Tm Group
- Host Application for various Plugins
- Plugins provide specific views and reports on data

SERVICES PLUGIN

SERVICE MANAGER PLUGIN

MESSAGES PLUGIN

LOCATIONS PLUGIN

APP CONFIG PLUGIN

exec

EISTCOCKPIT SERVICE

- RESTful JSON Webservice
- .NET-based, WCF
- Used by Client Applications
- Common Interface for CRUD
- Hosted on Server Infrastructure

DATA STORAGE

- SQLite Data Base
- Access via EistCockpitService

Abbildung 52 Systemübersicht



7.3.1.1 Allgemeine Anmerkungen

Die bidirektionalen Pfeile zwischen der Clientkomponente und der Servicekomponente deuten an, dass die Kommunikation zwischen den Komponenten in beide Richtungen verläuft. Die Richtung des Pfeils steht dabei für die Flussrichtung der Daten.

Wie der Beschriftung der Servicekomponente entnommen werden kann, bietet diese CRUD Funktionalität an; READ entspricht dem Pfeil in Richtung der Clientkomponenten (Daten werden zur Clientkomponente gesendet) während CREATE, UPDATE und DELETE dem Pfeil in Richtung der Servicekomponente entspricht (Daten werden an die Servicekomponente gesendet).

Jegliche Kommunikation mit der Servicekomponente wird von der Clientkomponente aus initiiert; d.h. es wird nach dem Polling-Prinzip vorgegangen. Die Servicekomponente antwortet lediglich auf Anfragen der Clientkomponente und initiiert selbständig keine Kommunikation.

7.3.1.2 Persistenz

Für die Persistenz der Daten wird eine SQLite Datenbank verwendet; der Zugriff darauf erfolgt ausschliesslich über den EistCockpitService (bzw. den darunter liegenden DAL).

Die Verwendung von SQLite bringt folgende Vorteile mit sich:

- Die Datenbank besteht aus einem einzelnen File, wodurch diese mit minimalem Aufwand kopiert werden kann. Dies ist ein wichtiger Punkt, denn die Daten sollen auch in nachfolgenden Dienstleistungen bzw. Wiederholungskursen zur Verfügung stehen.
- Die Software soll ohne grossen Installationsaufwand eingesetzt werden können; dies wird durch den Verzicht auf ein ausgewachsenes Datenbank-System (z.B. Microsoft SQL Server) sichergestellt.

7.3.1.3 Service

Clients greifen über den EistCockpitService auf die Daten zu. Der Service ist als RESTful JSON Webservice implementiert und stellt eine einheitliche CRUD API zur Verfügung.

Der Service wird auf dem Server des Systems ausgeführt.

7.3.1.4 Client

Die Client Applikation ist ein Rich Client, welcher mit WPF umgesetzt ist. Die Applikation dient als Host Umgebung für eine Vielzahl von Plugins, welche die effektiven Features implementieren.

Die Client Applikation wird auf den verschiedenen Arbeitsstationen ausgeführt und greift über das Netzwerk auf den Service zu.

7.3.1.5 Services Plugin

Das Service Plugin ist ein Host Plugin und listet die erfassten Dienstleistungen sowie deren Operationen hierarchisch in einer Baum-Darstellung auf.

Über dieses Plugin wird der Arbeitskontext für die verschiedenen Operation Plugins gesetzt bzw. ausgewählt.



7.3.1.6 Locations Plugin

Das Locations Plugin ist ein Operation Plugin und ermöglicht den Zugriff auf die hinterlegten Daten der Standorte, welche jeweils einer Operation zugewiesen sind.

Im Rahmen der Semesterarbeit implementiert dieses Plugin nur den Zugriff auf allgemeine Standort-Informationen. Im Rahmen der Bachelorarbeit wird der Funktions-Umfang beträchtlich erweitert werden (Personal, Verbindungen, Karten, etc.).

7.3.1.7 Messages Plugin

Das Messages Plugin ist ein Operation Plugin und bietet Zugriff auf die erfassten Nachrichten für eine Operation. Weiter werden umfangreiche Filter- und Such-Möglichkeiten angeboten.

7.3.1.8 App Config Plugin

Das App Config Plugin ist ein spezielles Host Plugin, welches die Verwaltung der Konfigurationsdaten der Applikation ermöglicht.

Die Konfigurationsdaten werden serverseitig in der Datenbank gehalten, da diese Einstellungen für alle Instanzen der Applikation gelten:

- Allgemeine Einstellungen
- Rollenverwaltung für Administrativ-Rechte
- Zugriffsverwaltung für die einzelnen Plugins

Der Zugriff auf dieses Plugin setzt Administrativ-Rechte voraus; diese werden automatisch der ActiveDirectory Gruppe "Administrators" gewährt.

7.3.1.9 Service Manager Plugin

Das Service Manager Plugin ist ein Host Plugin, welches die Erfassung von Dienstleistungen und deren Operationen ermöglicht.

Der Zugriff auf dieses Plugin setzt Administrativ-Rechte voraus.

7.3.2 Layers

Das Projekt verwendet die nachfolgend beschriebenen Layers. Es ist dabei zu beachten, dass sich die Schichten auf einen Server- und einen Client-Teil aufteilen.



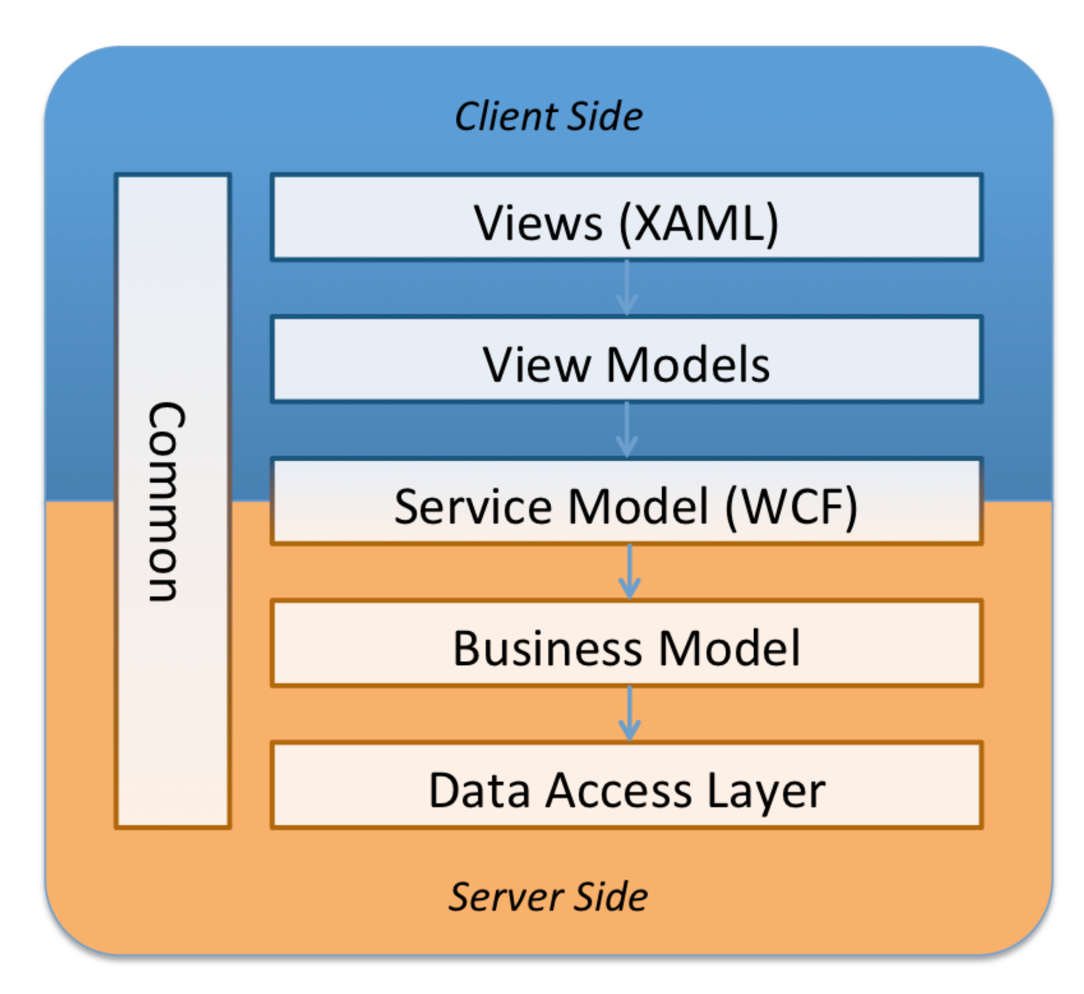


Abbildung 53 EistCockpit Layers

7.3.2.1 Views (XAML)

Der View Layer enthält UI Komponenten wie Fenster, Dialoge, User Controls, etc. Die Definition des User Interface wird in XAML (Extensible Application Markup Language) ausgeführt; unterstützende Funktionalität wird in sog. Code Behind Files in C# implementiert. Jeglicher Code in diesem Layer implementiert ausschliesslich UI-unterstützende Funktionalität, während spezifische Daten dem jeweiligen Viewmodel aus der darunterliegenden Schicht entnommen werden.

Durch die Trennung von UI- und Applikations-Logik wird die Kapselung erhöht und somit auch Testbarkeit der Software verbessert.

7.3.2.2 View Models

Der Viewmodel Layer hält die Daten für einen assoziierten View aus der darüberliegenden Schicht. Änderungen von Properties/Attributen im View Layer werden an das sog. Data Binding System, welches von Framework zur Verfügung gestellt wird, delegiert. Dieses stellt sicher, dass die Änderungen auch im entprechenden View Model umgesetzt werden; dies gilt auch in die Umgekehrte Richtung vom View Model zum View. Definitionsgemäss darf das View Model keinen Zugriff auf UI-Instanzen aus dem View Layer besitzen. Durch das Data Binding System wird die Kommunikation zwischen diesen Layern ermöglicht.

Wie bereits im vorherigen Abschnitt erwähnt, wird durch die Trennung zwischen View und View Model die Kapeslung erhöht und die Koppelung minimiert, was sich positiv auf die Qualität des Endprodukts auswirkt.



7.3.2.3 Service Model (WCF)

Der Service Model Layer stellt einen Service basierend auf WCF zur Verfügung. Mit Hilfe von WCF ist es möglich, Nachrichten zwischen zwei Service-Endpunkten *asynchron* auszutauschen. Im vorliegenden Fall ermöglicht der Service über eine REST Schnittstelle den Zugriff auf Daten im JSON Format; im Grunde stellt dieser Layer den Zugriff auf den Business Model Layer zur Verfügung.

Das Service Model unterteilt sich dabei auf einen Client-seitigen sowie einen Server-seitigen Teil:

- Clientseitig wird ein Service Interface implementiert, welches Anfragen mit Hilfe von WCF an die serverseitige Implementation weiterleitet. Die clientseitige Implementation wird durch den View Model Layer angesprochen, um benötigte Daten zu erhalten.
- Serverseitig wird das Service Interface ebenfalls implementiert; die Implementation greift dabei auf den darunterliegenden Business Model Layer zu, um die gewünschten Daten zu erhalten.

In WCF kommt standardmässig WSDL als Zugriffsschnittstelle und SOAP als Datenformat zum Einsatz. Die Verwendung von REST als Zugriffsschnittstelle und JSON als Datenformat bringt jedoch einige Vorteile gegenüber WSDL/SOAP mit sich:

- Der Datenzugriff via REST ist einfach, elegant und äusserst technologieunabhängig, da REST auf einfachen HTTP-Prinzipien aufgebaut ist. Der Rohdaten-Datenzugriff kann sogar über einen simplen Web Browser erfolgen.
- Das Datenformat JSON ist sehr kompakt und einfach lesbar (auch für Menschen). Zudem ist es weniger anfällig auf Fehler in der Formatierung der einzelnen Datenfelder. Die Kompaktheit des Formats resultiert in weniger Traffic bzw. Netzbelastung im LAN.
- EistCockpit wird zwar für eine Windows-Umgebung entwickelt und verwendet weitgehend Microsoft Technologien. Da die Daten aber letztendlich über REST/JSON abgegriffen werden, kann so auch ohne Probleme z.B. ein iOS Mobile Client oder eine Browser-basierte Webapplikation entwickelt werden.

7.3.2.4 Business Model

Der Business Model Layer stellt verschiedene Methoden zur Verfügung, welche die Businesslogik implementieren und CRUD Operationen auf dem Datenbestand ermöglichen. Der Zugriff auf diese Funktionalität geschieht dabei durch den darüberliegenden Service Model Layer.

Der Business Model Layer greift auf den darunterliegenden DAL (Data Access Layer) zu, um die gestellten Anfragen auszuführen. Dadurch wird die Business Logik von der verwendeten Datenbank entkoppelt.

7.3.2.5 Data Access Layer

Der DAL (Data Access Layer) abstrahiert den Zugriff auf den Persistenz Layer (z.B. Datenbank, XML Files, etc.). D.h. der DAL bietet Methoden an, welche grundlegende CRUD Operationen auf dem Datenbestand ermöglichen. Zu diesem Zweck bietet das .NET Framework unter der Bezeichnung ADO.NET sog. OR (Object Relation) Mapper an, welche basierend auf dem Schema einer relationalen Datenbank entsprechende Klassen generieren.



EistCockpit verwendet aus Gründen der Einfachheit und Portierbarkeit eine SQLite Datenbank, welche mit dem ADO.NET Adapter 'System.Data.SQLite'[25] angesprochen wird.

7.3.2.6 Common

Der Common Layer enthält diverse Klassen und Interfaces, welche von allen Layern verwendet werden.

Konkret fallen darunter:

- Entities & DTOs: Klassen für die Entities bzw. DTOs (Data Transfer Objects), da diese die Datenobjekte kapseln.
- Service Interface: Kommt Server- wie auch Client-seitig zum Einsatz.
- Plugin Interface: Wird von Plugins wie auch von der Host Applikation verwendet.
- Utilities: Utility Klassen, welche z.B. Validierungs-Funktionalität oder Konstanten zur Verfügung stellen.

7.3.3 Package Struktur

Die Package Struktur geht aus der Source Code Dokumentation hervor; für Details zu den einzelnen Packages bzw. deren Inhalt wird daher auf die Source Code Dokumentation verwiesen.

Nachfolgend erfolgt eine grobe hierarchische Übersicht der Packages von EistCockpit:

- **EistCockpit** – Root Package.
- **Business** – Der Business Layer implementiert die Business-Logik.
- **Common** – Enthält Elemente, welche von weiten Teilen von EistCockpit gemeinsam verwendet werden, z.B. Konstanten.
- **DTOs** – Definiert die Data Transfer Objects, welche vom Service an die Clients geschickt werden.
- **Contracts** – Definiert Interfaces & Contracts, welche im Zusammenhang mit der Plugin Architektur verwendet werden.
- **DAL** – Der Data Access Layer implementiert die Logik für den Datenzugriff.
- **Loader** – Enthält die EistCockpit Host Applikation.
- **Plugins** – Enthält die Packages für die einzelnen Plugins.
- **AppConfigPlugin** – Implementation des AppConfigPlugin.
- **LocationsPlugin** – Implementation des LocationsPlugin.
- **MessagesPlugin** – Implementation des MessagesPlugin.
- **ServiceManagerPlugin** – Implementation des ServiceManagerPlugin.
- **ServicesPlugin** – Implementation des ServicesPlugin.
- **Testing** – Enthält Unit Tests und Mockups für das Root Package.
- **ViewModels** – View Models der Host Applikation.
- **Views** – Views der Host Applikation sowie allgemein verwendbare User Controls.
- **Webservice** – Enthält die Implementation des Webservice.

---

[25] [url18]: System.Data.SQLite: About, http://system.data.sqlite.org/index.html/doc/trunk/www/index.wiki (05.06.2012)



7.3.4 Patterns

Während der Entwicklung von EistCockpit kamen diverse Entwicklungspatterns zum Einsatz. Nachfolgend wird eine Auswahl der verwendeten Patterns vorgestellt.

7.3.4.1 NULL Object Pattern

Das NULL Object Pattern hat zum Ziel, dass NULL Referenzen von Pointers durch ein existierendes, spezielles Objekt definiert werden. Anstatt dass Referenzen auf 'null' überprüft werden, wird nun überprüft, ob eine Referenz auf das wohl bekannte NULL Object zeigt.

Da das NULL Object definiert ist, werden dadurch Ausnahmezustände wie die berüchtigte "NULL Reference Exception" und die damit verbundenen Programmabstürze von vornherein verunmöglicht; eine Speicherverletzung kann beim Zugriff auf das NULL Object nicht stattfinden – eben weil es klar definiert ist.

EistCockpit verwendet das NULL Object Pattern im DAL: Wenn eine Entität mit einer spezifischen ID angefordert wird, welche aber auf der Datenbank nicht existiert, dann wird anstatt 'null' ein NULL Object von Typ der angeforderten Entität zurückgegeben.

Die NULL Objects sind in den Konstanten im Package 'Common' definiert.

7.3.4.2 Singleton Pattern

Das Singleton Pattern hat zum Ziel, dass exakt eine Instanz eines Objekts existiert. Verschiedene Teile der Software verwenden so immer die gleiche Instanz.

Während an der HSR eine regelrechte Verbannungstaktik für dieses Pattern gelehrt wird, hat es doch seine dedizierten Vorteile, weshalb der Einsatz dieses Patterns durchaus Sinn machen kann und legitim ist.[26]

Im Fall von EistCockpit wird das Singleton Pattern dazu verwendet, um einen einfachen Zugriff auf den aktuellen Dienstleistungs- bzw. Operations-Kontext für die verschiedenen Operation Plugins zu gewähren.

7.3.4.3 Command Pattern

Das Command Pattern abstrahiert das Event Handling insofern, als dass damit wiederverwendbare Kommandos/Befehle definiert werden können. Zudem steigert der Einsatz dieses Patterns die Testbarkeit der Software.

Im Rahmen von EistCockpit wird das Command Pattern, in enger Verbindung mit dem Model-View-ViewModel Konzept, innerhalb der ViewModels eingesetzt.

---

[26] Vergleichbare Beispiele aus der realen Welt sind z.B. die Handhabung einer Waffe, oder nur schon das Führen eines Fahrzeugs. Der Umgang mit diesen Gegenständen will gelernt sein, da sie bei falscher Anwendung oder aufgrund fehlender Erfahrung eine Gefahr für die Umwelt darstellen können. Natürlich ist die Gefahr gross, dass ein blutjunger Schmalspurprogrammierer beim Einsatz des Singleton Patterns Fehler begeht. Das bedeutet aber noch lange nicht, dass das Pattern durch erfahrene Entwickler keine Verwendung mehr finden darf.



EistCockpit

8 Prototypen

David Schöttl, Remo Waltenspül & Diego Steiner



8.1 Dokumenteninformation

Datum	Version	Änderung	Autor
05.06.2012	1.0	Erste Version des Dokuments	rwaltens
05.06.2012	1.1	Prototyp zu MEF Plugin dokumentiert	rwaltens
06.06.2012	1.2	Prototyp zu Datenbank dokumentiert	dsteiner
07.06.2012	1.3	Prototyp zu Webservice dokumentiert	dschöttl
08.06.2012	1.5	Review	rwaltens



8.2 Allgemein

8.2.1 Zweck des Dokumentes

Ziel dieses Dokumentes ist es, die entwickelten Prototypen näher zu erläutern, Dies beinhaltet eine grobe Architekturübersicht sowie Screenshots vom externen Design.

8.2.2 Gültigkeitsbereich

Dieses Dokument gilt als Grundlage des Projektes und ist daher über die gesamte Projektdauer gültig (20.02 bis 07.06.2012).

8.2.3 Definitionen und Abkürzungen

Siehe „Glossar“



8.3 Übersicht

8.3.1 Ziel

Bevor die Applikation entwickelt werden konnte, existierten noch verschiedene Unklarheiten betreffend der Architektur und dem Vorgehen. Dies stellten Risiken dar (Verweis Anhang Abschnitt Risikomanagement) die man mit spezifischen Prototypen entkräften wollte. Ein weiterer Grund für die Entwicklung von Prototypen war die einfachere Einarbeitung in neue Technologien und Frameworks anhand eines überschaubaren Testprojekts.

8.3.2 Durchführungszeitraum

- 04. April 2012 – 24. April 2012

8.3.3 Fazit

Mit den implementierten Prototypen konnte sichergestellt werden, dass die vom Projektteam beabsichtigen Architekturkonzepte innert nützlicher Zeit umgesetzt werden können.

Dabei konnten sämtliche Projektmitglieder viele neue Erfahrungen sammeln, die bei der anschliessenden Implementation einen schnelleren Einstieg erlaubten.



8.4 Prototypen

8.4.1 Webservice Prototyp

8.4.1.1 Zweck

Der Webservice Prototyp wurde als Proof-of-Concept erstellt, welcher aufzeigen soll, dass die Verwendung eines WCF REST/JSON Webservice zusammen mit einem WPF Client funktioniert.

Als Beispiel-Kontext wurde ein Gefechtsjournal (Combat Journal) gewählt: Der POC soll einige generierte Journal Einträge vom Service abrufen und die Details für den angewählten Eintrag anzeigen; um sicherzustellen, dass die Kommunikation auch vom Client zum Server funktioniert, wird, stellvertretend für die CRUD Operationen, eine Löschfunktion eingebaut.

8.4.1.2 Architektur

Die Projektstruktur des Webservice Prototypen umfasst drei Projekte:

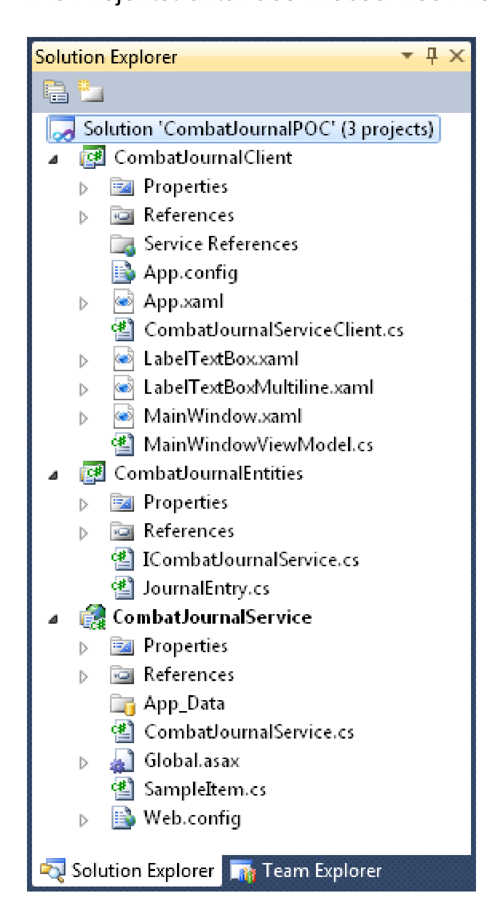


Abbildung 54 Projektstruktur 'CombatJournalPOC'

Projekt	Beschreibung
CombatJournalClient	Simple WPF Applikation, welche den Client für den Webservice darstellt.
CombatJournalEntities	Class Library, welche das Service Interface und die verwendete Entity (JournalEntry) implementiert.
CombatJournalService	WCF Webservice, welcher für REST/JSON konfiguriert ist und das Service Interface

implementiert.

Tabelle 58 Projektstruktur Prototyp Webservice

Das Service Projekt wurde mit dem, in VisualStudio unter Online Templates abrufbaren, WCF REST Service Project Template erstellt.

Die dargestellten Daten im UI von CombatJournalClient werden über Bindings propagiert.

8.4.1.3 Eingesetzte Technologien & Frameworks

Nachfolgend werden kurz alle Technologien aufgelistet, die für die Entwicklung des Webservice Prototyps nötig waren.

- .NET Framework 4.0, C#
- WCF, WPF
- REST, JSON

8.4.1.4 Externes Design

Die folgenden Abbildungen zeigen Screenshots vom WPF 'CombatJournalClient', sowie die über einen Web Browser abgerufenen Rohdaten (nach Löschung von Einträgen mittels WPF Client).

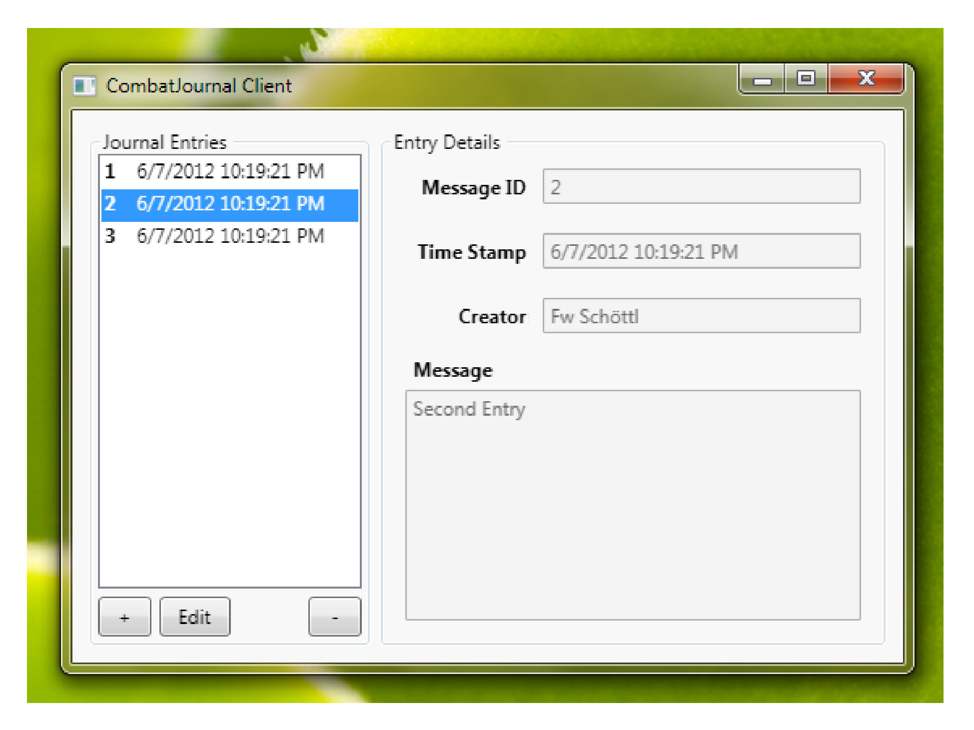


Abbildung 55 UI 'CombatJournalClient'

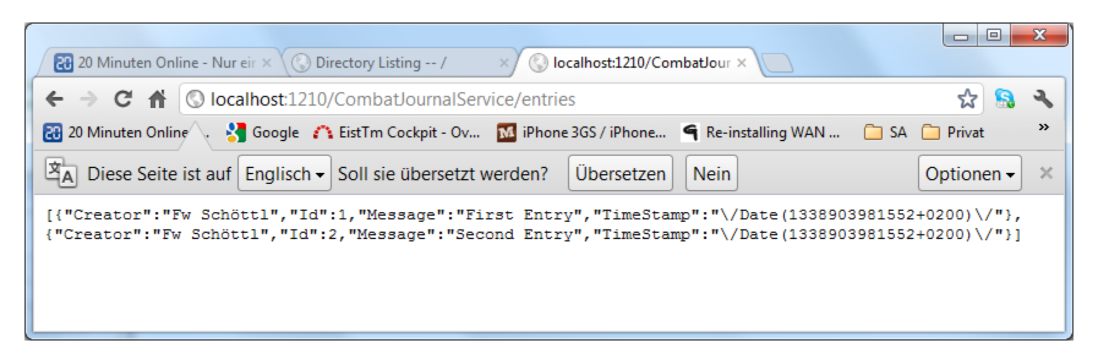


Abbildung 56 JSON Rohdaten, Eintrag '3' gelöscht.



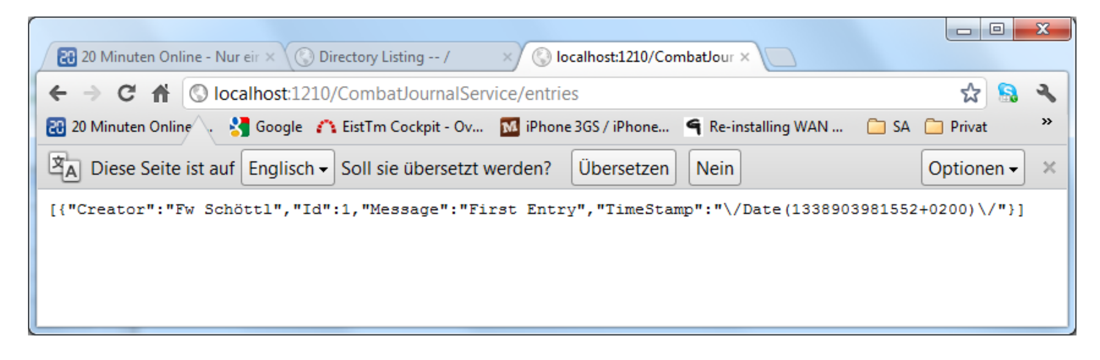


Abbildung 57 JSON Rohdaten, Eintrag '2' gelöscht.

8.4.1.5 Erkenntnisse

Während bis anhin nur mit .NET Webservices auf Basis von WSDL/SOAP gearbeitet wurde, konnte mit dem vorliegenden Prototypen die Arbeitsweise/Konfiguration für einen REST/JSON Webservice in Erfahrung gebracht werden.

Der Prototyp hat bewiesen, dass das Zusammenspiel zwischen WCF Webservice und WPF Prototyp gut funktioniert; nach anfänglichen Startschwierigkeiten traten keine signifikanten Probleme auf.

Da neben dem WCF Service auch ein WPF Client entwickelt wurde, konnten die Kenntnisse im Bereich WPF aufgefrischt werden. Ebenso wurde im Bereich WPF erkannt, dass das standardmässige '.NET Framework 4.0 Client Profile' für unsere Zwecke ungenügend ist und entsprechend auf '.NET Framework 4.0' gesetzt werden muss.

Da im Protoyp selbst erstellte Entities verwendet wurden (also keine, welche aus einer Datenbank generiert wurden), konnte zu diesem Zeitpunkt noch nicht erkannt werden, dass generierte Entities im Zusammenhang mit REST/JSON nicht funktionieren.

8.4.2 Datenbank Prototyp

8.4.2.1 Zweck

Der Datenbank Prototyp dient dazu den Umgang mit dem ADO .NET Framework zu verbessern. Weiter sollen die wichtigsten Tabellen bereits durch die automatische Objektrelationale Abbildung (O/R-Mapping) erzeugt werden, sowie die Anbindung an die SQLite Datenbank mittels eines Datenbankproviders realisiert werden.

8.4.2.2 Architektur

Die Architektur für den SQLite Prototyp beschränkt sich auf ein einzelnes Projekt, in welchem alles abgehandelt wird.



Projektstruktur

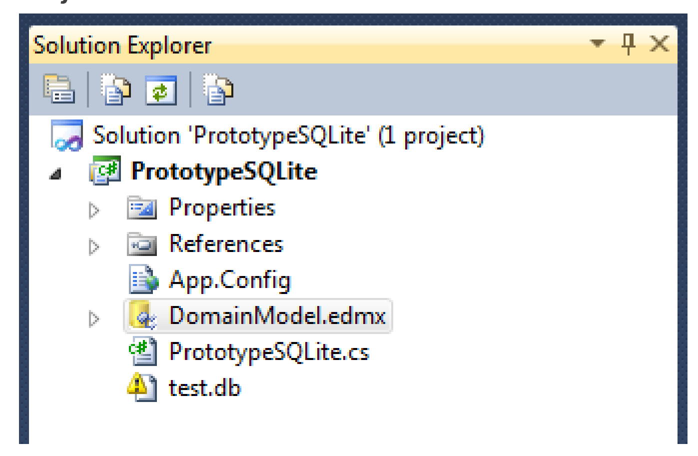


Abbildung 58: Projektstruktur SQLite -Prototyp

Item	Beschreibung
App.config	Definiert den Datenbankstring, mit welchem Diese angesprochen werden kann.
DomainModel.edmx	Deklariert alle Entities für das Entity Framework. Eis grossteil der für das EistCockpit benötigten Entitäten ist hier bereits definiert und kann mit wenig Aufwand in das Endprodukt übernommen werden.
PrototypeSQLite.cs	Testet die Funktionalität der SQLite-Datenbank und des Entity-Frameworks und bietet gleich auch Beispielcode für mögliche Zugriffe an.
Test.db	Die Beispieldatenbank für den Prototypen.

Tabelle 59 Projektstruktur Prototyp Datenbank



Datenmodell

Service
- Properties
  - id
  - name
  - description
  - from
  - to
- Navigation Properties
  - Operations

Operation
- Properties
  - id
  - name
  - type
  - service_id
- Navigation Properties
  - locations
  - Service
  - Phases

Phase
- Properties
  - id
  - name
  - description
  - order
  - operation_id
  - from
  - to
- Navigation Properties
  - TeardownConne...
  - SetupConnections
  - Operation
  - Achievments

Connection
- Properties
  - first_location_id
  - second_location...
  - setup_phase_id
  - teardown_phase...
- Navigation Properties
  - MasterLocation
  - SlaveLocation
  - TeardownPhase
  - SetupPhase

Location
- Properties
  - id
  - full_name
  - short_name
  - address
  - owner_address
  - owner_contact
  - person_responsi...
  - unit_id
  - coordinate_id
  - occupation_from
  - occupation_to
  - contact
  - operation_id
- Navigation Properties
  - SlaveConnections
  - MasterConnecti...
  - Coordinate
  - operations
  - Unit
  - ResponsiblePers...
  - locations_persons
  - Achievments

Coordinate
- Properties
  - id
  - latitude
  - longitude
  - altitude
  - ch1903
- Navigation Properties
  - Location

Unit
- Properties
  - id
  - name
  - parent_id
- Navigation Properties
  - Locations
  - Persons

Achievment
- Properties
  - id
  - description
  - type
  - datetime_should
  - datetime_is
  - phase_id
  - location_id
- Navigation Properties
  - Phase
  - Location

LocationPerson
- Properties
  - location_id
  - person_id
  - function
- Navigation Properties
  - locations
  - persons

Message
- Properties
  - id
  - sender_id
  - receipient_id
  - priority
  - message_type
  - text
  - created_by
  - created_at
  - updated_at
- Navigation Properties
  - Creator
  - Sender
  - Receipiant

Person
- Properties
  - id
  - rank
  - first_name
  - last_name
  - address
  - function
  - phone
  - remarks
  - status
  - unit_id
- Navigation Properties
  - LocationRespon...
  - locations_persons
  - CreatedMessages
  - SentMessages
  - ReceivedMessag...
  - Unit

Abbildung 59: Datenmodell SQLite-Prototyp



8.4.2.3 Eingesetzte Technologien & Frameworks

Nachfolgend werden kurz alle Technologien aufgelistet, die für die Entwicklung des Datenbank Prototyps nötig waren.

- Framework ADO .NET
- Datenbank Sqlite
- Datenbankprovider für Anbindung Sqlite an ADO .NET
- .NET C# 4.0

8.4.2.4 Erkenntnisse

Die SQLite-Implementation für das Entity-Framework eignet sich sehr gut für unsere Zwecke, obwohl die dezentrale Struktur der Datenbank-Files beim Deployment Schwierigkeiten verursachen könnte.

Die Integration ins Entity-Framework fühlt sich Nahtlos an, bisher sind keine Probleme bekannt.

8.4.3 MEF-Plugin

8.4.3.1 Zweck

Da es eine Anforderung war, die Applikation so zu entwerfen, dass sie einfach erweitert werden kann. Wurde im Internet nach bestehenden Frameworks für ein Plugin System Ausschau gehalten. Dieser Prototyp hat als Ziel die Verwendung des Microsoft Extensibility Frameworks (MEF) auf die Brauchbarkeit zu testen und zu prüfen ob die Anforderungen mit diesem Framework abgedeckt werden können.

8.4.3.2 Architektur

Projektstruktur

Aus der untenstehenden Abbildung (Verweis Abbildung 60: Projektstruktur MEF-Prototyp) geht hervor wie das Projekt aufgebaut ist.



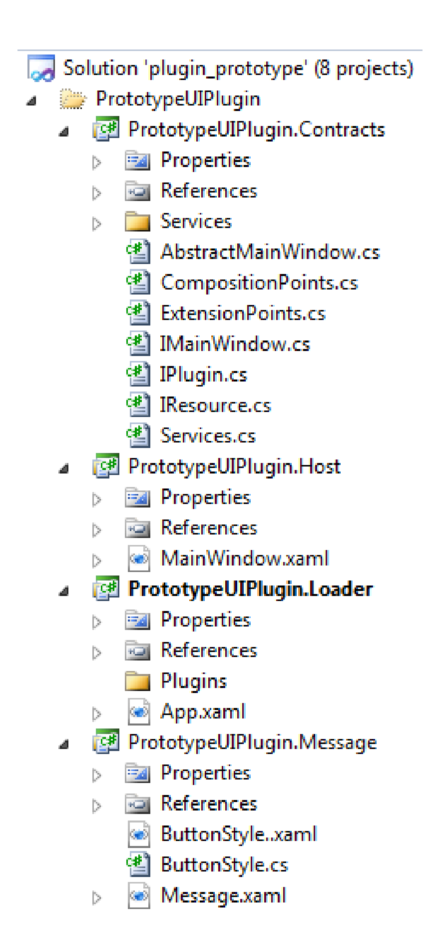


Abbildung 60: Projektstruktur MEF-Prototyp

Projekt	Beschreibung
Contracts	Enthält die wichtigsten Schnittstellendefinitionen für den Import von Plugins. Die zu ladenden Plugins müssen deshalb eine Referenz auf dieses Projekt enthalten.
Host	Die Host Applikation mit der in XAML definierten Benutzeroberfläche, welche durch das Loader Projekt geladen wird.
Loader	Der Loader stellt den Einstiegspunkt für die Applikation dar und ist verantwortlich für das Binden und Laden der Plugins.
Message	Bei diesem Projekt handelt es sich um ein Beispiel Plugin, welches den Importvertrag bzw. die Schnittstelle implementieren muss, damit es korrekt eingebunden werden kann. Grundsätzlich existieren zwei User Controls in diesem Projekt, eines wird für die Schaltfläche und das zweite für den Plugin Inhalt verwendet.

Tabelle 60 Projektstruktur Prototyp MEF Plugin

Sequenzdiagramm Laufzeitimport von Plugins

Das folgende Sequenzdiagramm zeigt den programmatischen Ablauf, der beim Start der Applikation durchgeführt wird. Es handelt sich dabei um eine leicht vereinfachte Darstellung, da einzelne für das Laden des Plugins nicht relevanten Anweisungen weggelassen wurden.



sd Dynamic View

:App
OnStartup(e)
Compose()
DirectoryCatalog(pluginPath)
directoryCatalog:DirectoryCatalog
AggregateCatalog()
catalog:AggregateCatalog
Plug-Ins werden gebunden
Catalogs.Add(directoryCatalog)
CompositionContainer(catalog,true)
container:CompositionContainer
CreateFileSystemWatcher()
FileSystemWatcher()
FileSystemWatche
ComposeParts(this)
OnChanged(source, e)
Refresh()
Window()
MainWindow:Window
OnImportsSatisfied()
refreshWindowWithAllPlugins()
Sobald neue .dll im Ordner hinzugekommen ist, wird EventHandler OnChanged ausgelöst

Abbildung 61: Sequenzdiagramm MEF-Prototyp

Import Annotations

MEF bietet sehr viele erweiterte Funktionen an. Für das Projekt EistCockpit wurden jedoch nur die Basisfunktionen eingesetzt. Die nachfolgenden Definitionen geben einen kurzen Überblick, für eine vollständige Dokumentation wird auf die msdn Seite http://msdn.microsoft.com/en-us/library/dd460648.aspx von Microsoft verwiesen.

Import

Die Import Annotation enthält einen optionalen Vertragsnamen sowie ein Interface, welches von genau einem Plugin erfüllt werden muss.

```
[Import(Services.Logging.LoggingService, typeof(ILoggingService)]
public ILoggingService logger { get; set; }
```

ImportMany

Der einzige Unterschied zu der Many Annotation liegt darin, dass ImportMany mehrere Plugins, die das Interface implementieren, einbinden kann.

```
[ImportMany(ExtensionPoints.Host.Views, typeof(ResourceDictionary)]
private IEnumerable<ResourceDictionary> Views { get; set; }
```



Export

Die Export Annotation markiert eine Klasse als für das MEF exportierbar. Wichtig ist dabei, dass das angegebene Interface vollständig implementiert wird.

```
[Export(ExtensionPoints.Host.Styles, typeof(ResourceDictionary))]
partial class ButtonStyle : ResourceDictionary
```

8.4.3.3 Eingesetzte Technologien & Frameworks

Nachfolgend werden kurz alle Technologien aufgelistet, die für die Entwicklung des MEF-Plugin Prototyps nötig waren.

- Plugin Framework MEF (Teil des PRISM Framework)
- .NET C#

8.4.3.4 Ablauf Plugin Integration zur Laufzeit

Vorbedingungen

- Interface von Import Anweisung muss mit Interface von zu exportierenden Klasse übereinstimmen.
- Falls in der Hauptapplikation ein Plugin über die Anweisung Import eingebunden wird, dürfen nicht mehrere Plugins diesen Vertrag erfüllen, ansonsten führt dies zu einem Fehler.

Ablauf

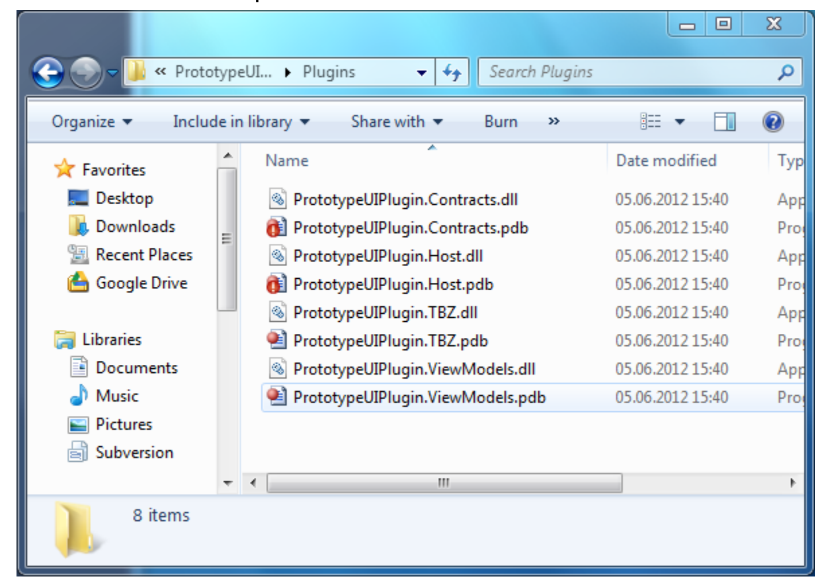
- Das Plugin wird in den vorgesehenen Plugin Ordner verschoben (Verweis siehe Abbildung Abbildung 62: Plugin Ordner)
- Der FileSystemWatcher erkennt die neu hinzugefügte dll
- Daraufhin ruft der FileSystemWatcher eine registrierte Callback-Methode auf
- Die Methode veranlasst die Klasse, welche für das Binden von Plugins zuständig ist, die Klassen neu zu importieren.
- Sobald der Import Vorgang beendet ist, werden Methoden aufgerufen um die User Controls aus den importierten Klassen in die Benutzeroberfläche zu integrieren.
- 

Abbildung 62: Plugin Ordner



8.4.3.5 Externes Design

Die folgenden Abbildungen zeigen Screenshots von der Testoberfläche, die verwendet wurde zum Prüfen, ob die Benutzeroberfläche zur Laufzeit um zusätzliche Plugins erweitert werden kann.

Zustand vor der Plugin Integration

Die Verweis Abbildung 63: Screenshot MEF-Prototyp 1 Macht den Zustand vor dem Laden des Plugin „Message“ ersichtlich.

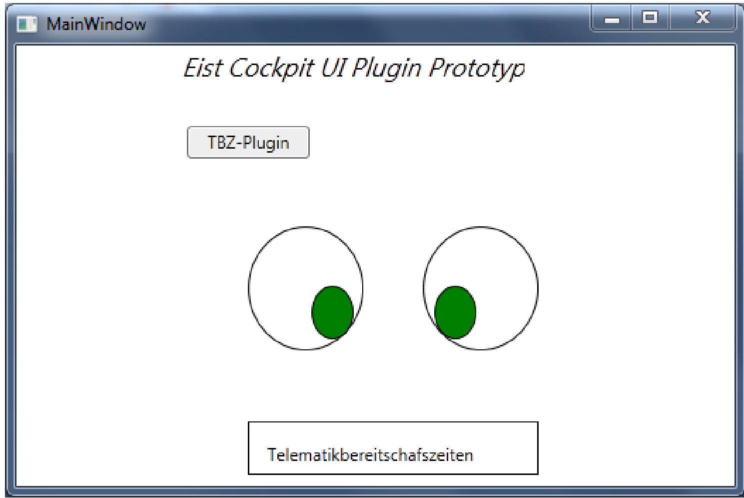


Abbildung 63: Screenshot MEF-Prototyp 1

Zustand nach der Plugin integration

Nach dem hinzufügen der Plugin abhängigen Dynamic Link Libraries (*.dll) zu dem definierten Plugin Ordner werden die neuen Klassen verglichen. Falls die Verträge in Form von Schnittstellen mit definiertem Import Anweisungen übereinstimmen, werden diese Klassen in das System eingebunden. Nach diesem Vorgang wird das User Control des neu geladenen Plugin „Message“ in der Benutzeroberfläche angezeigt (siehe Abbildung Abbildung 64: Screenshot MEF-Prototyp 2).

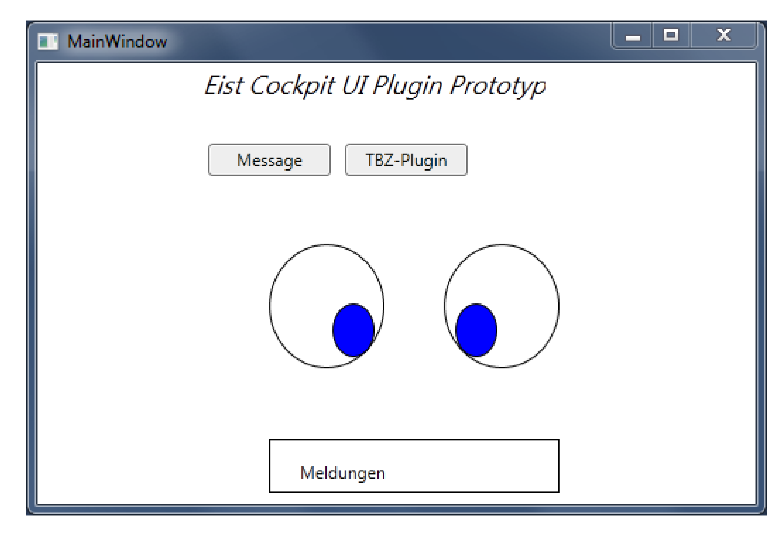


Abbildung 64: Screenshot MEF-Prototyp 2



8.4.3.6 Erkenntnisse

Die Einarbeitungszeit in die Basis Funktionen des MEF war ziemlich kurz, dementsprechend konnte schon nach kurzer Zeit der Code um Plugins erweitert werden. Eine gewisse Herausforderung war es jedoch eine Benutzeroberfläche zur Laufzeit um bestimmte Inhalte zu ergänzen. Um dies zu erreichen wurde zuerst versucht selber eine Klasse zu implementieren, die prüft ob ein Ordner neue zusätzliche Dateien enthält. Während der Implementation wurde aber die bereits bestehende Klasse FileSystemWatcher gefunden, welche die geforderte Funktionalität mitbringt.

Eine Schwierigkeit war auch die wenig aussagekräftige Fehlermeldung, die bei einem nicht erfolgreichen importieren von Plugins auftrat. Mit der Zeit wurden jedoch die Gründe für eine erscheinende Fehlermeldung ersichtlicher.



EistCockpit

9 Realisierung & Test

David Schöttl, Remo Waltenspül & Diego Steiner



9.1 Dokumenteninformation

Datum	Version	Änderung	Autor
04.06.2012	1.0	Erste Version des Dokuments	dsteiner
05.06.2012	1.1	Systemtests hinzugefügt	dsteiner
05.06.2012	1.2	1. Teil Codedokumentation erstellt	dsteiner
07.06.2012	1.3	Code Review dokumentiert	dsteiner
07.06.2012	1.5	Externes Design dokumentiert	rwaltens
07.06.2012	1.6	Kleinere Rechtschreibefehler behoben	rwaltens
08.06.2012	1.7	Review	dschöttl



9.2 Allgemein

9.2.1 Zweck des Dokumentes

Das Dokument Realisierung & Test dient dazu die Realisierungsphase sowie sämtliche durchgeführten Tests zu dokumentieren.

9.2.2 Gültigkeitsbereich

Dieses Dokument gilt als Grundlage des Projektes und ist daher über die gesamte Projektdauer gültig (20.02 bis 07.06.2012).

9.2.3 Definitionen und Abkürzungen

Siehe „Glossar".



9.3 Unit Tests

In diesem Abschnitt werden alle durchgeführten Unit Tests mittels Screenshots dargestellt. Dabei wurden nur Projekte getestet, die Methoden enthalten, welche einen hohen Reifegrad erfordern.

9.3.1 Sprint 2

Während dem Sprint 2 wurden die Hauptfunktionalitäten sowohl server-, wie auch clientseitig implementiert. Für den korrekten Zugriff auf die Datenbank musste sichergestellt werden, dass die Methoden auf dem Data Access Layer einwandfrei funktionieren. Folge dessen wurden in diesem 2. Sprint erst die DAL-Methoden getestet.

9.3.1.1 EistCockpit.Testing

DataAccessLayer

38 | 38 | 0 | 0 | 0

Test	Result
<EistCockpit.Testing> *(38 tests)*	Success
{ } EistCockpit.Testing *(38 tests)*	Success
UnitTestsDAL *(38 tests)*	Success
TestCoordinateWithID	Success
TestCreateCoordinate	Success
TestCreateFunction	Success
TestCreateLocation	Success
TestCreateLocationPerson	Success
TestCreateMessage	Success
TestCreateOperation	Success
TestCreatePerson	Success
TestCreateService	Success
TestCreateUnit	Success
TestDeleteCoordinateWithID	Success
TestDeleteFunctionWithID	Success
TestDeleteLocationPersonWithID	Success
TestDeleteLocationsPersonsWithLocationID	Success
TestDeleteLocationsPersonsWithPersonID	Success
TestDeleteLocationWithID	Success
TestDeleteMessageWithID	Success
TestDeleteOperationWithID	Success
TestDeletePersonWithID	Success
TestDeleteServiceWithID	Success
TestDeleteUnitWithID	Success
TestFunctionWithID	Success
TestLocationsPersonsForLocationID	Success
TestLocationsPersonsForPersonID	Success
TestLocationWithID	Success
TestMessageWithID	Success
TestOperationWithID	Success
TestPersonWithID	Success
TestServiceWithID	Success
TestUnitWithID	Success
TestUpdateCoordinate	Success
TestUpdateFunction	Success
TestUpdateLocation	Success
TestUpdateMessage	Success
TestUpdateOperation	Success
TestUpdatePerson	Success
TestUpdateService	Success
TestUpdateUnit	Success

Abbildung 65: Unit Tests DAL



9.3.2 Sprint 3

Umgehend nach Beginn des 3. Sprints wurden die Methoden der Plugins sowie der ServerOperationContext getestet.

9.3.2.1 EistCockpit.Testing

DataAccessLayer

50 | 50 | 0 | 0 | 0

Test	Result
<EistCockpit.Testing> *(50 tests)*	Success
{ } EistCockpit.Testing *(50 tests)*	Success
UnitTestsDAL *(50 tests)*	Success
TestAdminRoleWithID	Success
TestConfigurationWithID	Success
TestCoordinateWithID	Success
TestCreateAdminRole	Success
TestCreateConfiguration	Success
TestCreateCoordinate	Success
TestCreateFunction	Success
TestCreateLocation	Success
TestCreateLocationPerson	Success
TestCreateMessage	Success
TestCreateOperation	Success
TestCreatePerson	Success
TestCreatePluginAccess	Success
TestCreateService	Success
TestCreateUnit	Success
TestDeleteAdminRoleWithID	Success
TestDeleteConfigurationWithID	Success
TestDeleteCoordinateWithID	Success
TestDeleteFunctionWithID	Success
TestDeleteLocationPersonWithID	Success
TestDeleteLocationsPersonsWithLocationID	Success
TestDeleteLocationsPersonsWithPersonID	Success
TestDeleteLocationWithID	Success
TestDeleteMessageWithID	Success
TestDeleteOperationWithID	Success
TestDeletePersonWithID	Success
TestDeletePluginAccessWithID	Success
TestDeleteServiceWithID	Success
TestDeleteUnitWithID	Success
TestFunctionWithID	Success
TestLocationsPersonsForLocationID	Success
TestLocationsPersonsForPersonID	Success
TestLocationWithID	Success
TestMessageWithID	Success
TestOperationWithID	Success
TestPersonWithID	Success
TestPluginAccessWithID	Success
TestServiceWithID	Success
TestUnitWithID	Success
TestUpdateAdminRole	Success
TestUpdateConfiguration	Success
TestUpdateCoordinate	Success
TestUpdateFunction	Success
TestUpdateLocation	Success
TestUpdateMessage	Success
TestUpdateOperation	Success
TestUpdatePerson	Success
TestUpdatePluginAccess	Success
TestUpdateService	Success
TestUpdateUnit	Success

Abbildung 66Unit Tests DAL



ServiceOperationContextTest

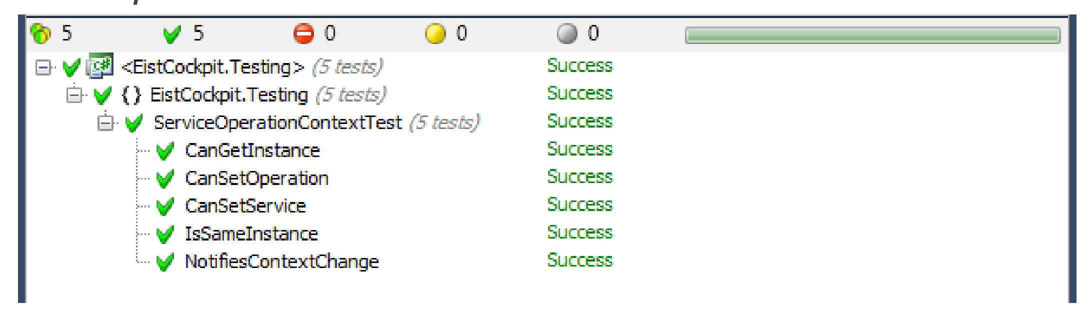


Abbildung 67: Unit Test ServiceOperationContext

9.3.2.2 EistCockpit.Plugins.LocationsPlugin.Test

LocationUnitTest

34 34 0 0 0

<EistCockpit.LocationsPlugin.Test> (34 tests) Success
{} EistCockpit.LocationsPlugin.Test (34 tests) Success
LocationUnitTest (34 tests) Success
AddressDisplayNoErrorMessage Success
AddressValidationCorrect Success
CanClearErrorDictionary Success
CanCloneLocation Success
CanDiscard Success
CanLoadLocation Success
CanReloadLocation Success
CanSaveNewLocation Success
CanSelectLocation Success
CanUpdateValidLocation Success
ChangeOfValidityOfLocations Success
ContactDisplayNoErrorMessage Success
ContactValidationCorrect Success
CorrectStringRepresentation Success
EqualsWorksCorrect Success
FromDisplayNoErrorMessage Success
FromValidationCorrect Success
FullNameDisplayNoErrorMessage Success
FullNameValidationCorrect Success
GetSelectedLocationReturnOneSelectedLocation Success
HasChangedPropertyCorrectValue Success
HasCorrespondingAttributes Success
HasCurrentAndOriginalDTO Success
IsNewRecordCorrectResult Success
IsValidOnlyIfAllRelationsSet Success
LoadsRelations Success
OwnerAddressDisplayNoErrorMessage Success
OwnerAddressValidationCorrect Success
OwnerContactDisplayNoErrorMessage Success
OwnerContactValidationCorrect Success
ShortNameDisplayNoErrorMessage Success
ShortNameValidationCorrect Success
ToDisplayNoErrorMessage Success
ToValidationCorrect Success

Abbildung 68: Unit Test Location



CoordinateUnitTest

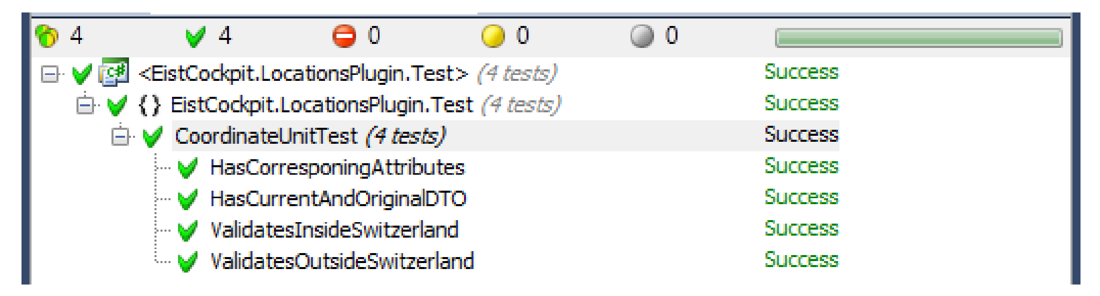


Abbildung 69: Unit Tests Coordinates

ShowLocationViewModelUnitTest

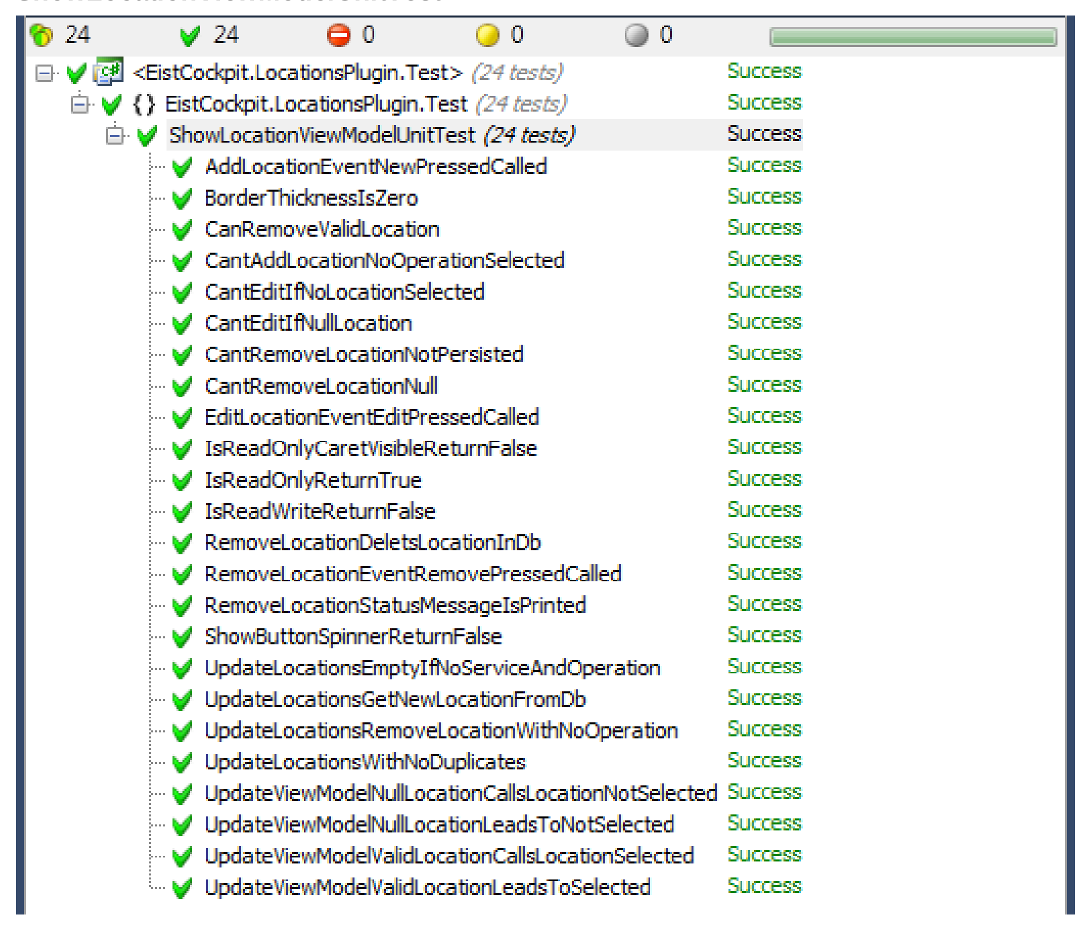


Abbildung 70: Unit Tests ShowLocationViewModel



EditLocationViewModelUnitTest

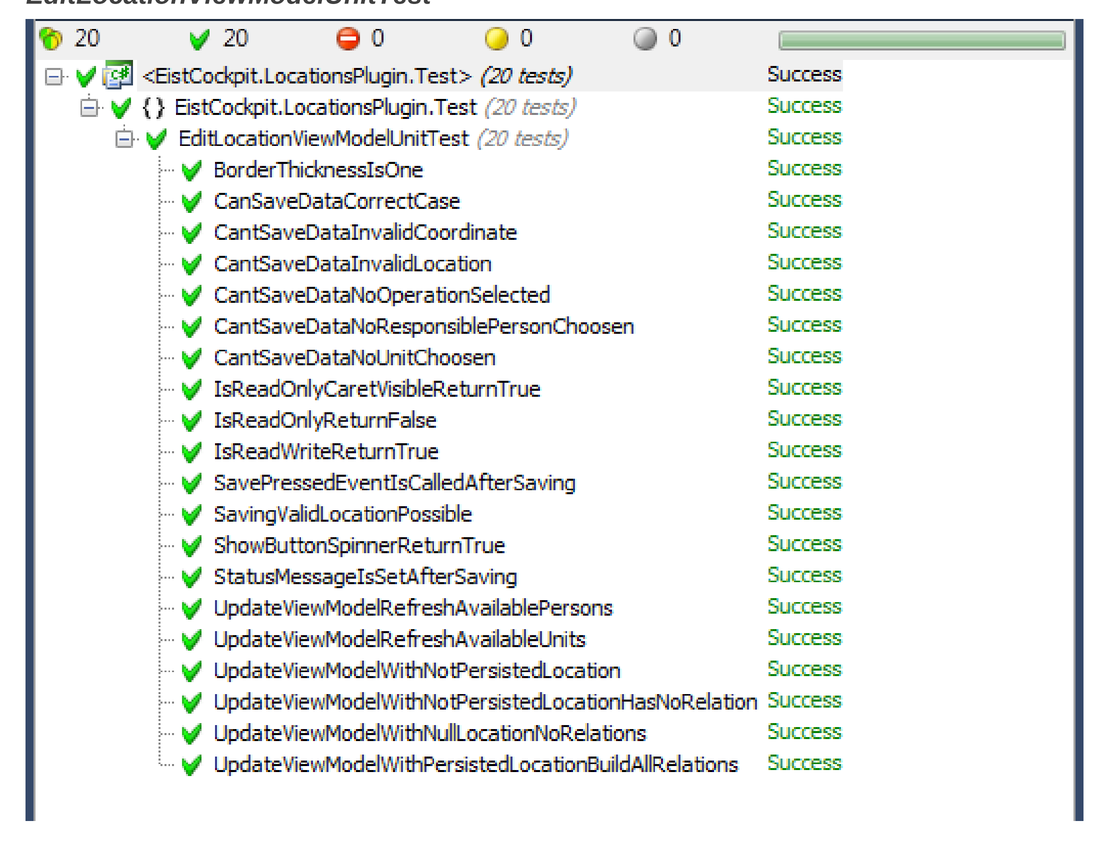


Abbildung 71: Unit Tests EditLocationViewModel



9.3.2.3 EistCockpit.Plugins.MessagesPlugin.Test

MessageTest

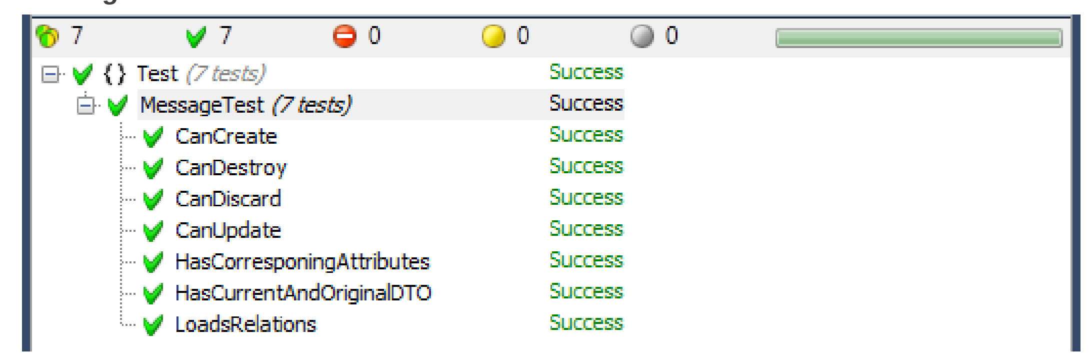


Abbildung 72: Unit Tests Message

MessagesCollectionTest

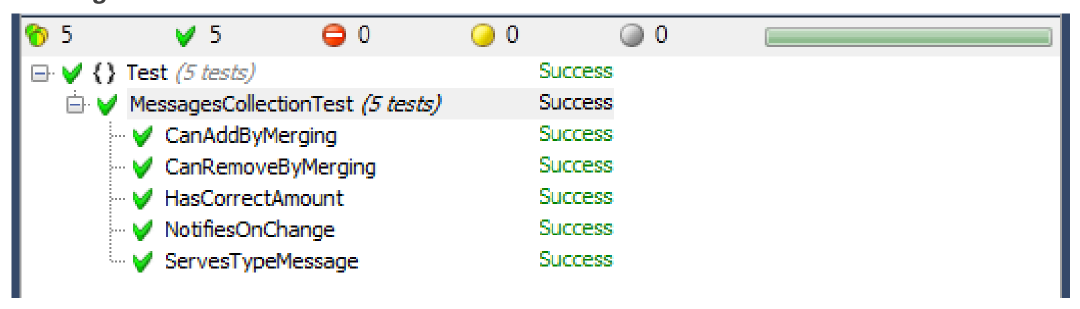


Abbildung 73: Unit Tests MessageCollection

MessageViewModelTest

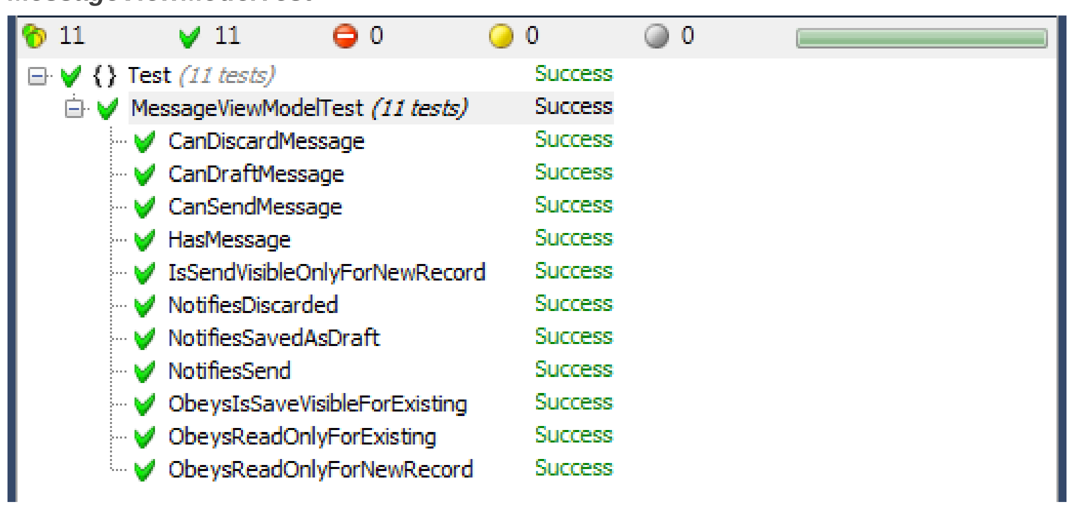


Abbildung 74: Unit Tests MessageViewModel

MessagesViewModelTest

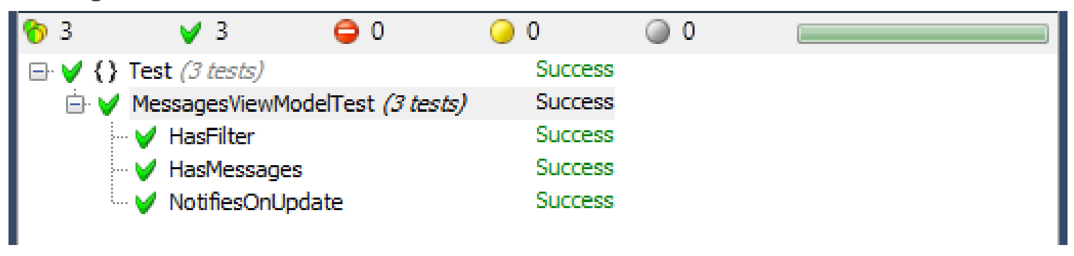


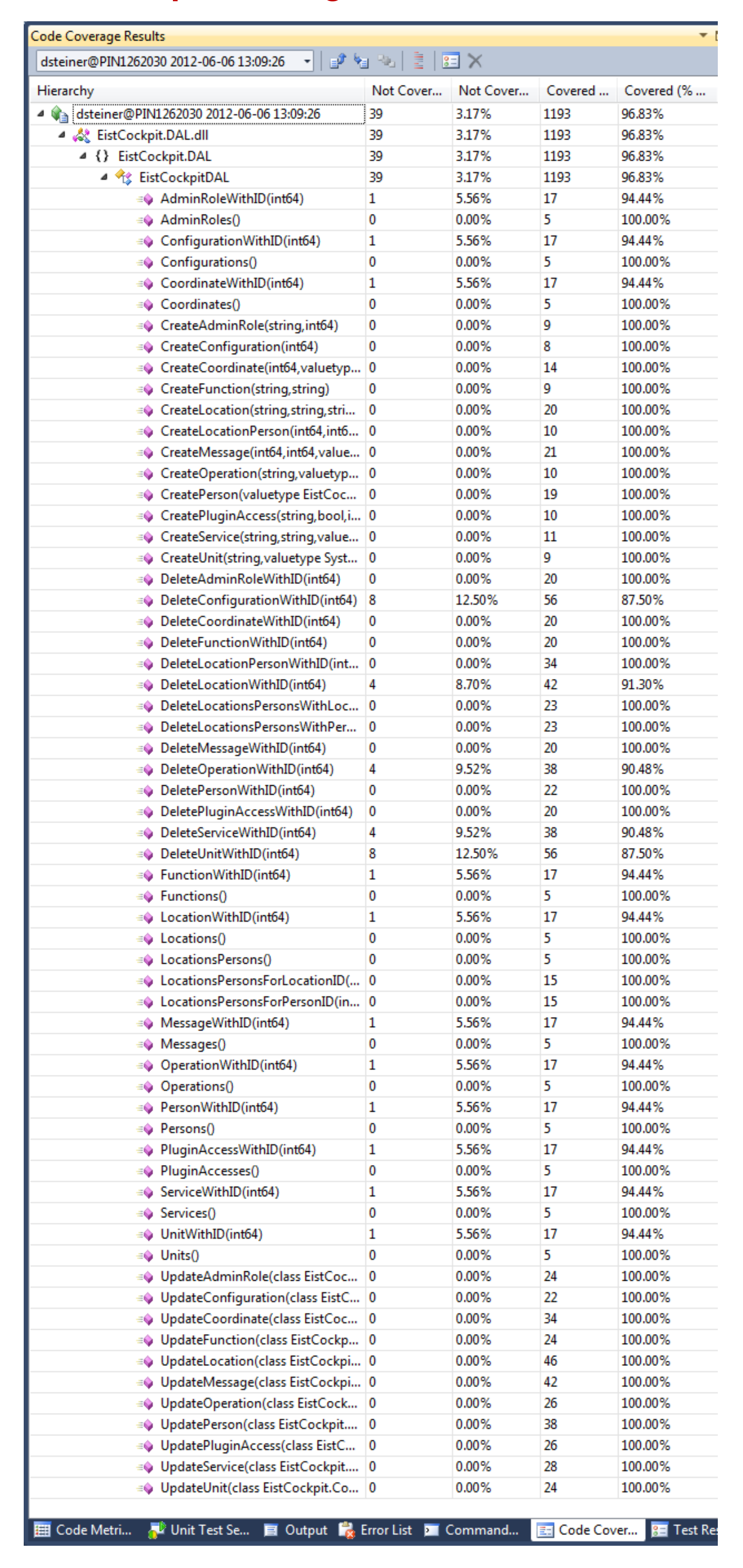
Abbildung 75: Unit Tests MessagesViewModel



9.3.3 Testabdeckung

Wie aus den nicht funktionalen Anforderungen hervorgeht (Verweis Abschnitt 5.5 Nichtfunktionale Anforderungen) soll im Durchschnitt die Testabdeckung bei den Plugins über 70% und bei dem DAL über 90% liegen. Anhand der folgenden Screenshots wird ersichtlich, dass die geforderten Werte erfüllt werden konnten.

9.3.3.1 EistCockpit.Testing



Code Coverage Results

dsteiner@PIN1262030 2012-06-06 13:09:26

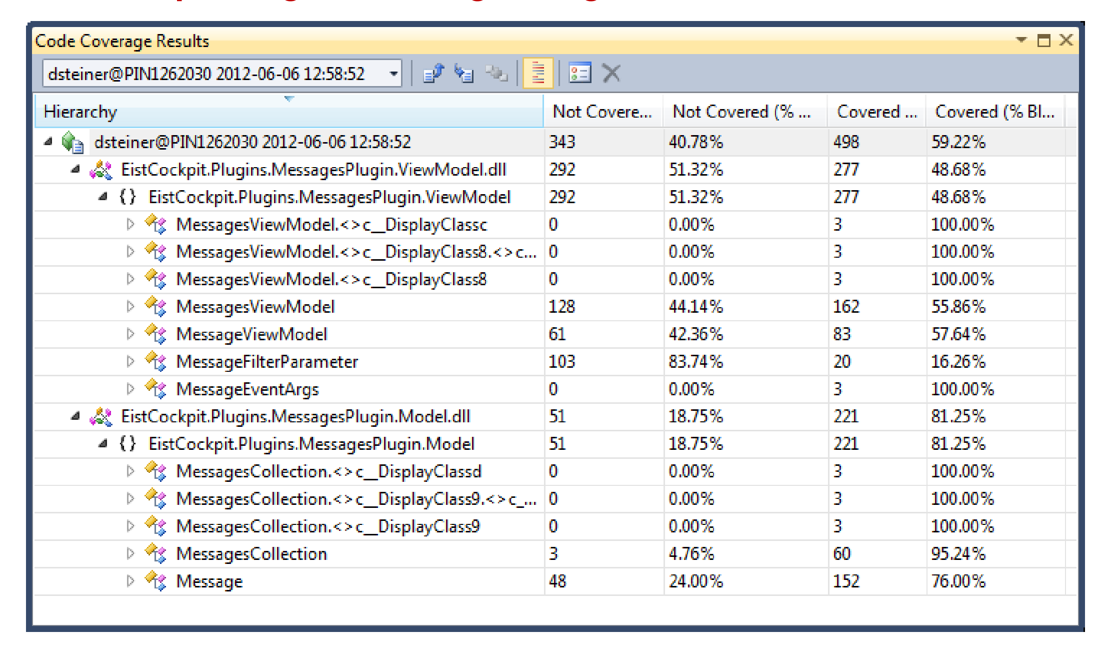
Hierarchy	Not Cover...	Not Cover...	Covered ...	Covered (% ...
dsteiner@PIN1262030 2012-06-06 13:09:26	39	3.17%	1193	96.83%
EistCockpit.DAL.dll	39	3.17%	1193	96.83%
{ } EistCockpit.DAL	39	3.17%	1193	96.83%
EistCockpitDAL	39	3.17%	1193	96.83%
AdminRoleWithID(int64)	1	5.56%	17	94.44%
AdminRoles()	0	0.00%	5	100.00%
ConfigurationWithID(int64)	1	5.56%	17	94.44%
Configurations()	0	0.00%	5	100.00%
CoordinateWithID(int64)	1	5.56%	17	94.44%
Coordinates()	0	0.00%	5	100.00%
CreateAdminRole(string,int64)	0	0.00%	9	100.00%
CreateConfiguration(int64)	0	0.00%	8	100.00%
CreateCoordinate(int64,valuetyp...	0	0.00%	14	100.00%
CreateFunction(string,string)	0	0.00%	9	100.00%
CreateLocation(string,string,stri...	0	0.00%	20	100.00%
CreateLocationPerson(int64,int6...	0	0.00%	10	100.00%
CreateMessage(int64,int64,value...	0	0.00%	21	100.00%
CreateOperation(string,valuetyp...	0	0.00%	10	100.00%
CreatePerson(valuetype EistCoc...	0	0.00%	19	100.00%
CreatePluginAccess(string,bool,i...	0	0.00%	10	100.00%
CreateService(string,string,value...	0	0.00%	11	100.00%
CreateUnit(string,valuetype Syst...	0	0.00%	9	100.00%
DeleteAdminRoleWithID(int64)	0	0.00%	20	100.00%
DeleteConfigurationWithID(int64)	8	12.50%	56	87.50%
DeleteCoordinateWithID(int64)	0	0.00%	20	100.00%
DeleteFunctionWithID(int64)	0	0.00%	20	100.00%
DeleteLocationPersonWithID(int...	0	0.00%	34	100.00%
DeleteLocationWithID(int64)	4	8.70%	42	91.30%
DeleteLocationsPersonsWithLoc...	0	0.00%	23	100.00%
DeleteLocationsPersonsWithPer...	0	0.00%	23	100.00%
DeleteMessageWithID(int64)	0	0.00%	20	100.00%
DeleteOperationWithID(int64)	4	9.52%	38	90.48%
DeletePersonWithID(int64)	0	0.00%	22	100.00%
DeletePluginAccessWithID(int64)	0	0.00%	20	100.00%
DeleteServiceWithID(int64)	4	9.52%	38	90.48%
DeleteUnitWithID(int64)	8	12.50%	56	87.50%
FunctionWithID(int64)	1	5.56%	17	94.44%
Functions()	0	0.00%	5	100.00%
LocationWithID(int64)	1	5.56%	17	94.44%
Locations()	0	0.00%	5	100.00%
LocationsPersons()	0	0.00%	5	100.00%
LocationsPersonsForLocationID(...	0	0.00%	15	100.00%
LocationsPersonsForPersonID(in...	0	0.00%	15	100.00%
MessageWithID(int64)	1	5.56%	17	94.44%
Messages()	0	0.00%	5	100.00%
OperationWithID(int64)	1	5.56%	17	94.44%
Operations()	0	0.00%	5	100.00%
PersonWithID(int64)	1	5.56%	17	94.44%
Persons()	0	0.00%	5	100.00%
PluginAccessWithID(int64)	1	5.56%	17	94.44%
PluginAccesses()	0	0.00%	5	100.00%
ServiceWithID(int64)	1	5.56%	17	94.44%
Services()	0	0.00%	5	100.00%
UnitWithID(int64)	1	5.56%	17	94.44%
Units()	0	0.00%	5	100.00%
UpdateAdminRole(class EistCoc...	0	0.00%	24	100.00%
UpdateConfiguration(class EistC...	0	0.00%	22	100.00%
UpdateCoordinate(class EistCoc...	0	0.00%	34	100.00%
UpdateFunction(class EistCockp...	0	0.00%	24	100.00%
UpdateLocation(class EistCockpi...	0	0.00%	46	100.00%
UpdateMessage(class EistCockpi...	0	0.00%	42	100.00%
UpdateOperation(class EistCock...	0	0.00%	26	100.00%
UpdatePerson(class EistCockpit....	0	0.00%	38	100.00%
UpdatePluginAccess(class EistC...	0	0.00%	26	100.00%
UpdateService(class EistCockpit....	0	0.00%	28	100.00%
UpdateUnit(class EistCockpit.Co...	0	0.00%	24	100.00%

Code Metri... | Unit Test Se... | Output | Error List | Command... | Code Cover... | Test Re

Abbildung 76: Testabdeckung DAL



9.3.3.2 EistCockpit.Plugins.MessagesPlugin.Test



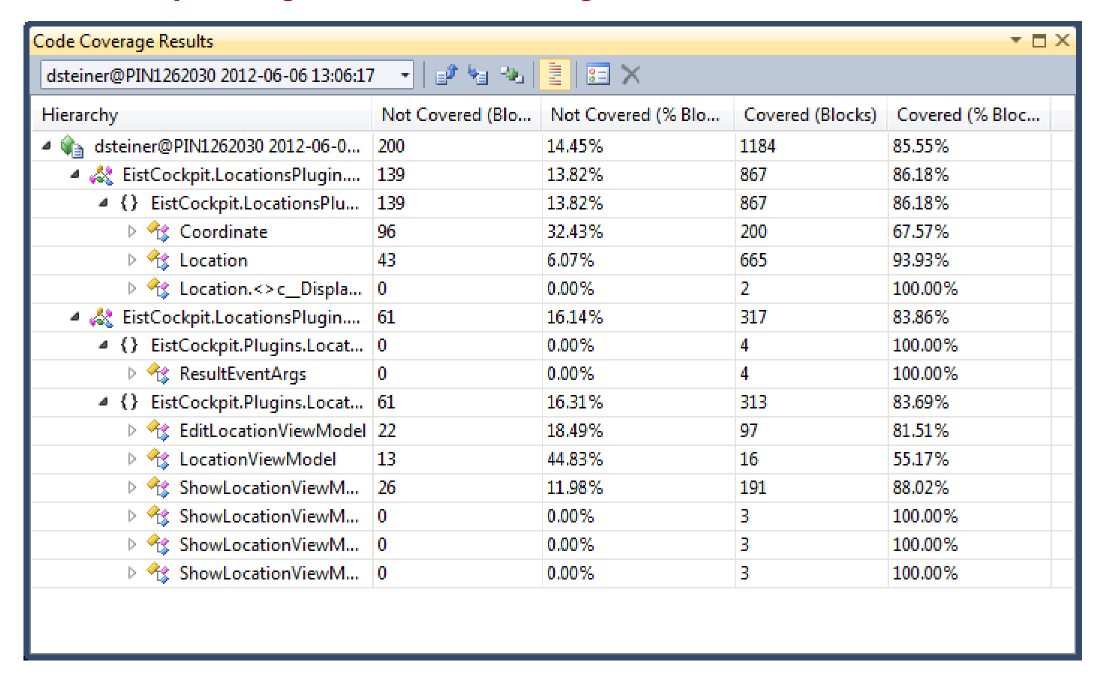
Code Coverage Results

dsteiner@PIN1262030 2012-06-06 12:58:52

Hierarchy	Not Covere...	Not Covered (% ...	Covered ...	Covered (% Bl...
dsteiner@PIN1262030 2012-06-06 12:58:52	343	40.78%	498	59.22%
EistCockpit.Plugins.MessagesPlugin.ViewModel.dll	292	51.32%	277	48.68%
{} EistCockpit.Plugins.MessagesPlugin.ViewModel	292	51.32%	277	48.68%
MessagesViewModel.<>c__DisplayClassc	0	0.00%	3	100.00%
MessagesViewModel.<>c__DisplayClass8.<>c...	0	0.00%	3	100.00%
MessagesViewModel.<>c__DisplayClass8	0	0.00%	3	100.00%
MessagesViewModel	128	44.14%	162	55.86%
MessageViewModel	61	42.36%	83	57.64%
MessageFilterParameter	103	83.74%	20	16.26%
MessageEventArgs	0	0.00%	3	100.00%
EistCockpit.Plugins.MessagesPlugin.Model.dll	51	18.75%	221	81.25%
{} EistCockpit.Plugins.MessagesPlugin.Model	51	18.75%	221	81.25%
MessagesCollection.<>c__DisplayClassd	0	0.00%	3	100.00%
MessagesCollection.<>c__DisplayClass9.<>c_...	0	0.00%	3	100.00%
MessagesCollection.<>c__DisplayClass9	0	0.00%	3	100.00%
MessagesCollection	3	4.76%	60	95.24%
Message	48	24.00%	152	76.00%

Abbildung 77: Testabdeckung Messages

9.3.3.3 EistCockpit.Plugins.LocationsPlugin.Test



Code Coverage Results

dsteiner@PIN1262030 2012-06-06 13:06:17

Hierarchy	Not Covered (Blo...	Not Covered (% Blo...	Covered (Blocks)	Covered (% Bloc...
dsteiner@PIN1262030 2012-06-0...	200	14.45%	1184	85.55%
EistCockpit.LocationsPlugin....	139	13.82%	867	86.18%
{} EistCockpit.LocationsPlu...	139	13.82%	867	86.18%
Coordinate	96	32.43%	200	67.57%
Location	43	6.07%	665	93.93%
Location.<>c__Displa...	0	0.00%	2	100.00%
EistCockpit.LocationsPlugin....	61	16.14%	317	83.86%
{} EistCockpit.Plugins.Locat...	0	0.00%	4	100.00%
ResultEventArgs	0	0.00%	4	100.00%
{} EistCockpit.Plugins.Locat...	61	16.31%	313	83.69%
EditLocationViewModel	22	18.49%	97	81.51%
LocationViewModel	13	44.83%	16	55.17%
ShowLocationViewM...	26	11.98%	191	88.02%
ShowLocationViewM...	0	0.00%	3	100.00%
ShowLocationViewM...	0	0.00%	3	100.00%
ShowLocationViewM...	0	0.00%	3	100.00%

Abbildung 78: Testabdeckung Locations



9.4 Systemtests

Resultate

Für die Systemtest wird das Naheliegenste der folgenden Resultate ausgewählt:

- Ok Test erfolgreich
- Failed Test gescheitert
- NI Nicht implementiert



9.4.1 Sprint 1 (14.05.2012)

#	Titel	Beschreibung	Resultat	Testperson
1	Applikation starten	Die Applikation kann gestartet werden	Ok	Dsteiner
2	Webservice starten	Der Webservice startet und im Browser ist die Übersicht angezeigt	Ok	Dschoet
3	Clientplugins laden	Die in den Plugins komplilierten DLLs werden geladen und im entsprechenden Bereich angezeigt	Ok	Rwaltens
4	Clientplugins auswählen	Im dafür vorgesehenen Bereich können die Plugins ausgewählt werden	Ok	Rwaltens
5	Hostplugins anzeigen	In dem dafür vorgesehenen Bereich werden die geladenen Hostplugins angezeigt	NI	Dsteiner
6	Hostplugins auswählen	Die Hostplugins können ausgewählt werden	NI	Dsteiner
7	Ausgewählter Service anzeigen	Wenn ein Service ausgewählt wird, wird in der Statusleiste Diese angezeigt	NI	Dsteiner
8	Operation anzeigen	Im Serviceplugin wird eine Übersicht der Operationen unter den entsprechenden Dienstleistungen angezeigt	NI	Dsteiner
9	Operation auswählen	Im Serviceplugin kann eine Operation ausgewählt werden	NI	Dsteiner
10	Ausgewählte Operation anzeigen	Wenn eine Operation ausgewählt wird, wird in der Statusleiste Diese angezeigt	NI	dsteiner
11	Messages anzeigen	Im MessagesPlugin wird eine Übersicht der Meldungen angezeigt	NI	Dsteiner
12	MessageFilter anzeigen	Im MessagesPlugin kann der Filter ein- und ausgeblendet werden	NI	Dsteiner
13	MessagesFilter anwenden	Im MessagesPlugin kann der Filter auf die Meldungen angewandt werden	NI	Dsteiner
14	Messages sortieren	Im MessagesPlugin können die Meldungen nach allen Spalten sortiert werden	NI	Dsteiner
15	Message anzeigen	Im MessagesPlugin kann eine einzelne Nachricht angezeigt werden	NI	Dsteiner
16	Message erfassen	Im MessagesPlugin kann eine neue Meldung erfasst werden	NI	Dsteiner
17	Message speichern	Im MessagesPlugin kann eine Meldung gespeichert werden	NI	Dsteiner
18	Message als Draft speichern	Im MessagesPlugin kann eine Meldung als Entwurf gespeichert werden	NI	Dsteiner
19	Messages refreshen	Im MessagesPlugin werden die neue Meldungen in der Übersicht angezeigt	NI	Dsteiner
20	Locations anzeigen	Im LocationsPlugin wird eine Übersicht der Standorte angezeigt	NI	Rwaltens
21	Location anzeigen	Im LocationsPlugin wird nach auswahl eines Standortes Dieser angezeigt	NI	Rwaltens
22	Location erfassen	Im LocationsPlugin kann ein neuer Standort erfasst werden	NI	Rwaltens
23	Location bearbeiten	Im LocationsPlugin kann ein ausgewählter Standort bearbeitet werden	NI	Rwaltens
24	Location speichern	Im LocationsPlugin kann ein bearbeiteter Standort gespeichert werden	NI	Rwaltens
25	Locations refreshen	Im LocationsPlugin werden neue Standorte in der Übersicht angezeigt	NI	Dsteiner
26	Die Applikation kann beendet werden		NI	Rwaltens

Tabelle 61 Systemtests nach Sprint 1



9.4.2 Sprint 2 (31.05.2012)

#	Titel	Beschreibung	Resultat	Testperson
1	Applikation starten	Die Applikation kann gestartet werden	Ok	Dsteiner
2	Webservice starten	Der Webservice startet und im Browser ist die Übersicht angezeigt	Ok	Dschoet
3	Clientplugins laden	Die in den Plugins komplilierten DLLs werden geladen und im entsprechenden Bereich angezeigt	Ok	Rwaltens
4	Clientplugins auswählen	Im dafür vorgesehenen Bereich können die Plugins ausgewählt werden	Ok	Rwaltens
5	Hostplugins anzeigen	In dem dafür vorgesehenen Bereich werden die geladenen Hostplugins angezeigt	NI	Dsteiner
6	Hostplugins auswählen	Die Hostplugins können ausgewählt werden	NI	Dsteiner
7	Ausgewählter Service anzeigen	Wenn ein Service ausgewählt wird, wird in der Statusleiste Diese angezeigt	NI	Dsteiner
8	Operation anzeigen	Im Serviceplugin wird eine Übersicht der Operationen unter den entsprechenden Dienstleistungen angezeigt	Ok	Dsteiner
9	Operation auswählen	Im Serviceplugin kann eine Operation ausgewählt werden	Ok	Dsteiner
10	Ausgewählte Operation anzeigen	Wenn eine Operation ausgewählt wird, wird in der Statusleiste Diese angezeigt	NI	dsteiner
11	Messages anzeigen	Im MessagesPlugin wird eine Übersicht der Meldungen angezeigt	Ok	Dsteiner
12	MessageFilter anzeigen	Im MessagesPlugin kann der Filter ein- und ausgeblendet werden	Ok	Dsteiner
13	MessagesFilter anwenden	Im MessagesPlugin kann der Filter auf die Meldungen angewandt werden	Failed	Dsteiner
14	Messages sortieren	Im MessagesPlugin können die Meldungen nach allen Spalten sortiert werden	NI	Dsteiner
15	Message anzeigen	Im MessagesPlugin kann eine einzelne Nachricht angezeigt werden	Ok	Dsteiner
16	Message erfassen	Im MessagesPlugin kann eine neue Meldung erfasst werden	Ok	Dsteiner
17	Message speichern	Im MessagesPlugin kann eine Meldung gespeichert werden	Ok	Dsteiner
18	Message als Draft speichern	Im MessagesPlugin kann eine Meldung als Entwurf gespeichert werden	NI	Dsteiner
19	Messages refreshen	Im MessagesPlugin werden die neue Meldungen in der Übersicht angezeigt	NI	Dsteiner
20	Locations anzeigen	Im LocationsPlugin wird eine Übersicht der Standorte angezeigt	Ok	Rwaltens
21	Location anzeigen	Im LocationsPlugin wird nach auswahl eines Standortes Dieser angezeigt	Ok	Rwaltens
22	Location erfassen	Im LocationsPlugin kann ein neuer Standort erfasst werden	NI	Rwaltens
23	Location bearbeiten	Im LocationsPlugin kann ein ausgewählter Standort bearbeitet werden	NI	Rwaltens
24	Location speichern	Im LocationsPlugin kann ein bearbeiteter Standort gespeichert werden	NI	Rwaltens
25	Locations refreshen	Im LocationsPlugin werden neue Standorte in der Übersicht angezeigt	NI	Rwaltens
26	Die Applikation kann beendet werden		Ok	Dsteiner

Tabelle 62 Systemtests nach Sprint 2



9.4.3 Sprint 3 (04.06.2012)

#	Titel	Beschreibung	Resultat	Testperson
1	Applikation starten	Die Applikation kann gestartet werden	Ok	Dsteiner
2	Webservice starten	Der Webservice startet und im Browser ist die Übersicht angezeigt	Ok	Dschoet
3	Clientplugins laden	Die in den Plugins komplilierten DLLs werden geladen und im entsprechenden Bereich angezeigt	Ok	Rwaltens
4	Clientplugins auswählen	Im dafür vorgesehenen Bereich können die Plugins ausgewählt werden	Ok	Rwaltens
5	Hostplugins anzeigen	In dem dafür vorgesehenen Bereich werden die geladenen Hostplugins angezeigt	Ok	Dsteiner
6	Hostplugins auswählen	Die Hostplugins können ausgewählt werden	Ok	Dsteiner
7	Ausgewählter Service anzeigen	Wenn ein Service ausgewählt wird, wird in der Statusleiste Diese angezeigt	Ok	Dsteiner
8	Operation anzeigen	Im Serviceplugin wird eine Übersicht der Operationen unter den entsprechenden Dienstleistungen angezeigt	Ok	Dsteiner
9	Operation auswählen	Im Serviceplugin kann eine Operation ausgewählt werden	Ok	Dsteiner
10	Ausgewählte Operation anzeigen	Wenn eine Operation ausgewählt wird, wird in der Statusleiste Diese angezeigt	ok	dsteiner
11	Messages anzeigen	Im MessagesPlugin wird eine Übersicht der Meldungen angezeigt	Ok	Dsteiner
12	MessageFilter anzeigen	Im MessagesPlugin kann der Filter ein- und ausgeblendet werden	Ok	Dsteiner
13	MessagesFilter anwenden	Im MessagesPlugin kann der Filter auf die Meldungen angewandt werden	Ok	Dsteiner
14	Messages sortieren	Im MessagesPlugin können die Meldungen nach allen Spalten sortiert werden	Ok	Dsteiner
15	Message anzeigen	Im MessagesPlugin kann eine einzelne Nachricht angezeigt werden	Ok	Dsteiner
16	Message erfassen	Im MessagesPlugin kann eine neue Meldung erfasst werden	Ok	Dsteiner
17	Message speichern	Im MessagesPlugin kann eine Meldung gespeichert werden	Ok	Dsteiner
18	Message als Draft speichern	Im MessagesPlugin kann eine Meldung als Entwurf gespeichert werden	Ok	Dsteiner
19	Messages refreshen	Im MessagesPlugin werden die neue Meldungen in der Übersicht angezeigt	Ok	Dsteiner
20	Locations anzeigen	Im LocationsPlugin wird eine Übersicht der Standorte angezeigt	Ok	Rwaltens
21	Location anzeigen	Im LocationsPlugin wird nach auswahl eines Standortes Dieser angezeigt	Ok	Rwaltens
22	Location erfassen	Im LocationsPlugin kann ein neuer Standort erfasst werden	Ok	Rwaltens
23	Location bearbeiten	Im LocationsPlugin kann ein ausgewählter Standort bearbeitet werden	Ok	Rwaltens
24	Location speichern	Im LocationsPlugin kann ein bearbeiteter Standort gespeichert werden	Ok	Rwaltens
25	Locations refreshen	Im LocationsPlugin werden neue Standorte in der Übersicht angezeigt	Ok	Rwaltens
26	Die Applikation kann beendet werden		Ok	Dsteiner

Tabelle 63Systemtests nach Sprint 3



9.5 Usability Tests

Es konnten keine Usabilitytest durchgeführt werden.



9.6 Codedokumentation

9.6.1 Autoren

Die Autorenschaft sämtlicher SourceCode-Dateien des EistCockpits ist gemeinschaftlich und aus diesem Grund nicht in jeder Datei separat deklariert.

9.6.2 SourceCode Dokumentation

Der SourceCode wird mittels Doxygen realisiert und generiert. Dafür müssen im Code die von Doxygen [27]vorgeschriebenen Kommentarregeln eingehalten werden.

9.6.3 Metriken

Für die Metriken wird das in VisualStudio 2010 integrierte Metriktool verwendet. Aus der Metrikanalyse geht hervor, dass alle Indikatoren im grünen Bereich liegen.

[27] Doxygen, http://www.stack.nl/~dimitri/doxygen/manual.html (08.06.2012)



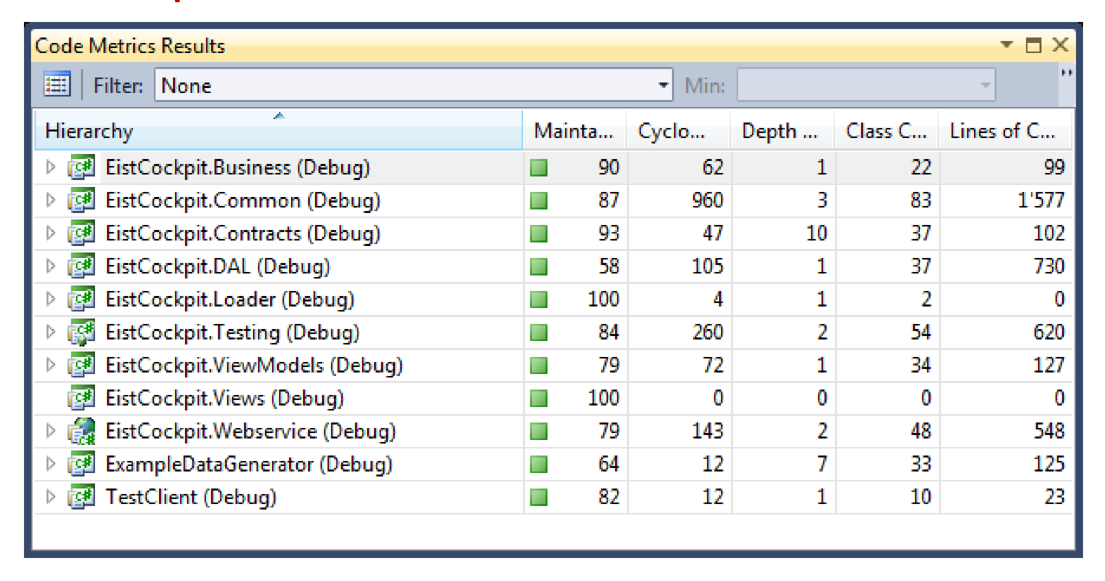
9.6.3.1 Lines of Code

Namespace	LOC
EistCockpit.Business	99
EistCockpit.Common	1577
EistCockpit.Contracts	102
EistCockpit.DAL	730
EistCockpit.Loader	0
EistCockpit.Testing	620
EistCockpit.ViewModels	127
EistCockpit.Views	0
EistCockpit.Webservice	548
EistCockpit.Plugins.MessagesPlugin.Loader	10
EistCockpit.Plugins.MessagesPlugin.Model	132
EistCockpit.Plugins.MessagesPlugin.Test	145
EistCockpit.Plugins.MessagesPlugin.View	30
EistCockpit.Plugins.MessagesPlugin.ViewModel	281
EistCockpit.Plugins.LocationsPlugin.Loader	9
EistCockpit.Plugins.LocationsPlugin.Model	427
EistCockpit.Plugins.LocationsPlugin.Test	404
EistCockpit.Plugins.LocationsPlugin.View	11
EistCockpit.Plugins.LocationsPlugin.ViewModel	282
EistCockpit.Plugins.ServicesPlugin.View	11
EistCockpit.Plugins.ServicesPlugin.ViewModel	18
EistCockpit.Plugins.ServicesPlugin.Loader	8
EistCockpit.Plugins.AppConfigPlugin.Loader	17
EistCockpit.Plugins.AppConfigPlugin.View	0
EistCockpit.Plugins.AppConfigPlugin.ViewModel	398
Applikation	**4817**
Test	**1169**
Gesamt	**5986**

Tabelle 64 Auswertung Codezeilen



9.6.3.2 EistCockpit



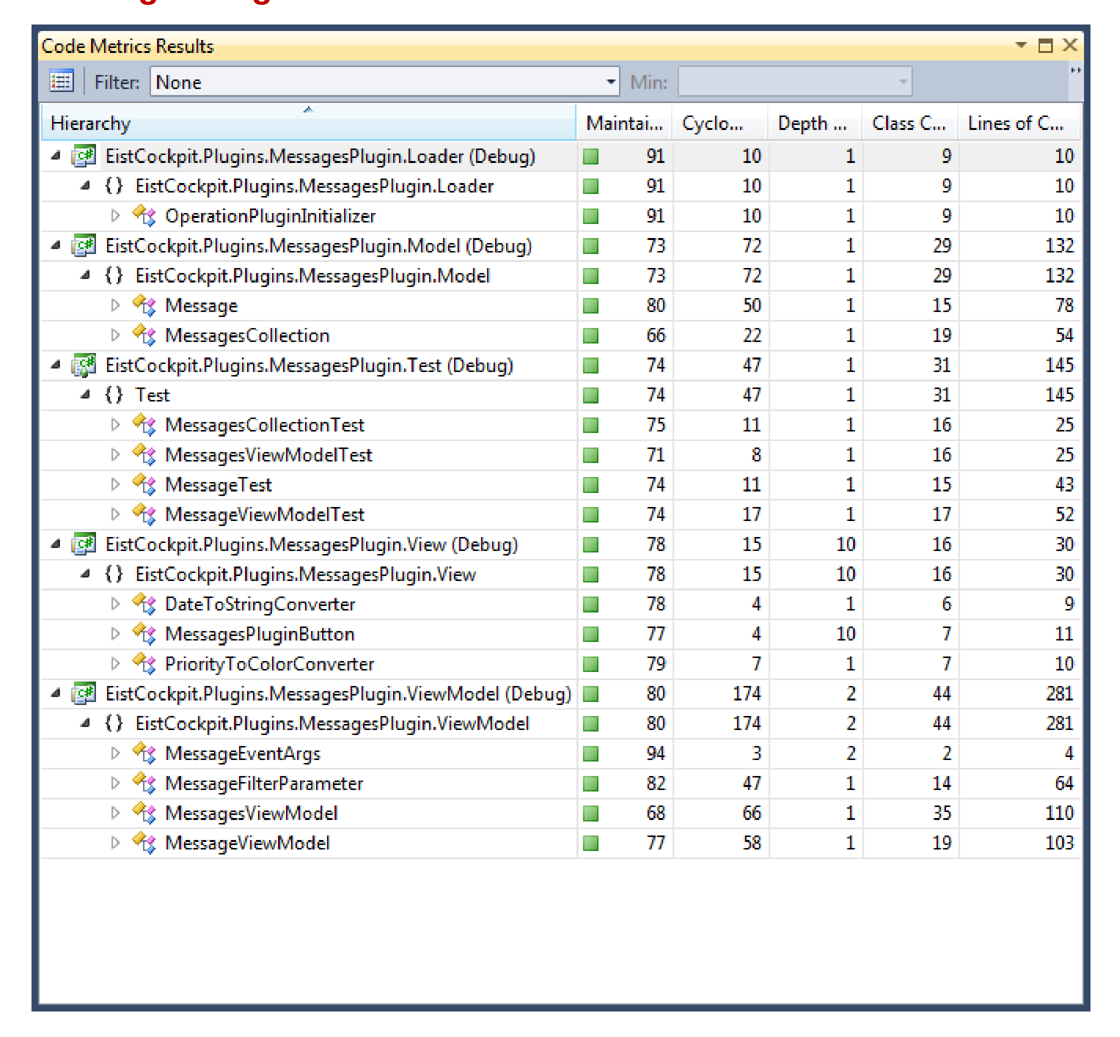
Code Metrics Results

Filter: None Min:

Hierarchy	Mainta...	Cyclo...	Depth ...	Class C...	Lines of C...
EistCockpit.Business (Debug)	90	62	1	22	99
EistCockpit.Common (Debug)	87	960	3	83	1'577
EistCockpit.Contracts (Debug)	93	47	10	37	102
EistCockpit.DAL (Debug)	58	105	1	37	730
EistCockpit.Loader (Debug)	100	4	1	2	0
EistCockpit.Testing (Debug)	84	260	2	54	620
EistCockpit.ViewModels (Debug)	79	72	1	34	127
EistCockpit.Views (Debug)	100	0	0	0	0
EistCockpit.Webservice (Debug)	79	143	2	48	548
ExampleDataGenerator (Debug)	64	12	7	33	125
TestClient (Debug)	82	12	1	10	23

Abbildung 79: Metrics EistCockpit

9.6.3.3 MessagesPlugin



Code Metrics Results

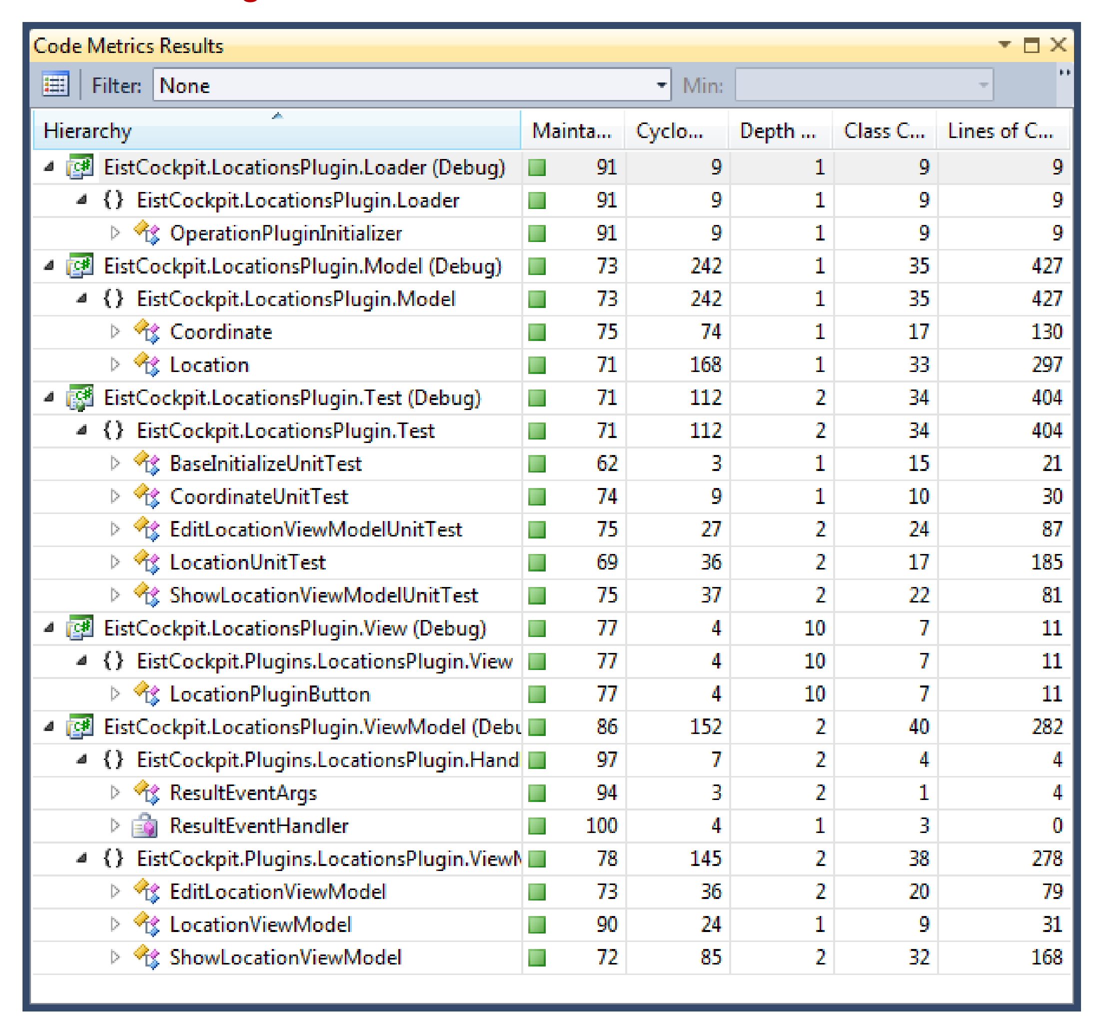
Filter: None Min:

Hierarchy	Maintai...	Cyclo...	Depth ...	Class C...	Lines of C...
EistCockpit.Plugins.MessagesPlugin.Loader (Debug)	91	10	1	9	10
{ } EistCockpit.Plugins.MessagesPlugin.Loader	91	10	1	9	10
OperationPluginInitializer	91	10	1	9	10
EistCockpit.Plugins.MessagesPlugin.Model (Debug)	73	72	1	29	132
{ } EistCockpit.Plugins.MessagesPlugin.Model	73	72	1	29	132
Message	80	50	1	15	78
MessagesCollection	66	22	1	19	54
EistCockpit.Plugins.MessagesPlugin.Test (Debug)	74	47	1	31	145
{ } Test	74	47	1	31	145
MessagesCollectionTest	75	11	1	16	25
MessagesViewModelTest	71	8	1	16	25
MessageTest	74	11	1	15	43
MessageViewModelTest	74	17	1	17	52
EistCockpit.Plugins.MessagesPlugin.View (Debug)	78	15	10	16	30
{ } EistCockpit.Plugins.MessagesPlugin.View	78	15	10	16	30
DateToStringConverter	78	4	1	6	9
MessagesPluginButton	77	4	10	7	11
PriorityToColorConverter	79	7	1	7	10
EistCockpit.Plugins.MessagesPlugin.ViewModel (Debug)	80	174	2	44	281
{ } EistCockpit.Plugins.MessagesPlugin.ViewModel	80	174	2	44	281
MessageEventArgs	94	3	2	2	4
MessageFilterParameter	82	47	1	14	64
MessagesViewModel	68	66	1	35	110
MessageViewModel	77	58	1	19	103

Abbildung 80: Metrics MessagesPlugin



9.6.3.4 LocationsPlugin



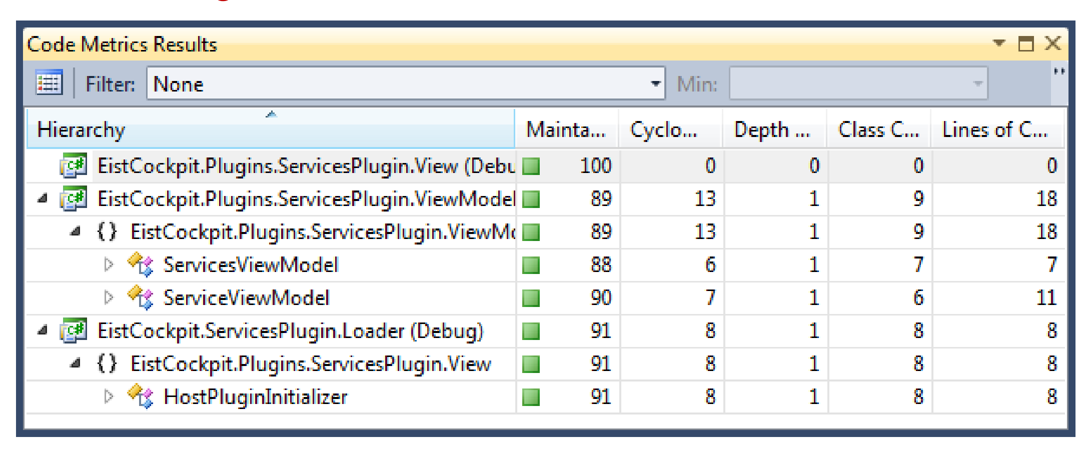
Code Metrics Results

Filter: None

Hierarchy	Mainta...	Cyclo...	Depth ...	Class C...	Lines of C...
EistCockpit.LocationsPlugin.Loader (Debug)	91	9	1	9	9
EistCockpit.LocationsPlugin.Loader	91	9	1	9	9
OperationPluginInitializer	91	9	1	9	9
EistCockpit.LocationsPlugin.Model (Debug)	73	242	1	35	427
EistCockpit.LocationsPlugin.Model	73	242	1	35	427
Coordinate	75	74	1	17	130
Location	71	168	1	33	297
EistCockpit.LocationsPlugin.Test (Debug)	71	112	2	34	404
EistCockpit.LocationsPlugin.Test	71	112	2	34	404
BaseInitializeUnitTest	62	3	1	15	21
CoordinateUnitTest	74	9	1	10	30
EditLocationViewModelUnitTest	75	27	2	24	87
LocationUnitTest	69	36	2	17	185
ShowLocationViewModelUnitTest	75	37	2	22	81
EistCockpit.LocationsPlugin.View (Debug)	77	4	10	7	11
EistCockpit.Plugins.LocationsPlugin.View	77	4	10	7	11
LocationPluginButton	77	4	10	7	11
EistCockpit.LocationsPlugin.ViewModel (Debu	86	152	2	40	282
EistCockpit.Plugins.LocationsPlugin.Hand	97	7	2	4	4
ResultEventArgs	94	3	2	1	4
ResultEventHandler	100	4	1	3	0
EistCockpit.Plugins.LocationsPlugin.ViewM	78	145	2	38	278
EditLocationViewModel	73	36	2	20	79
LocationViewModel	90	24	1	9	31
ShowLocationViewModel	72	85	2	32	168

Abbildung 81: Metrics LocationsPlugin

9.6.3.5 ServicesPlugin



Code Metrics Results

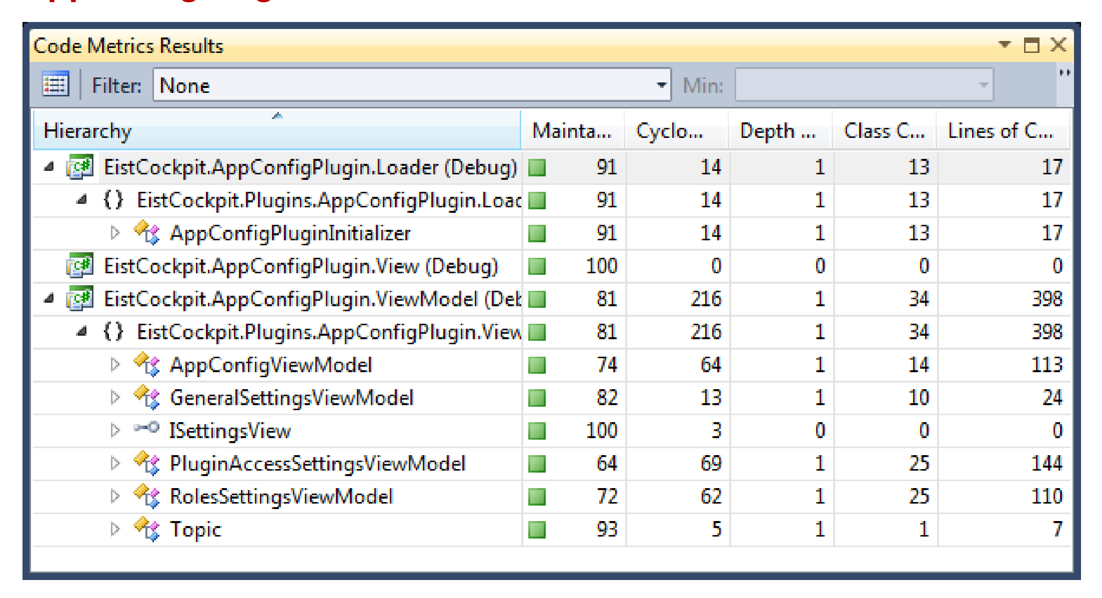
Filter: None

Hierarchy	Mainta...	Cyclo...	Depth ...	Class C...	Lines of C...
EistCockpit.Plugins.ServicesPlugin.View (Debu	100	0	0	0	0
EistCockpit.Plugins.ServicesPlugin.ViewModel	89	13	1	9	18
EistCockpit.Plugins.ServicesPlugin.ViewMo	89	13	1	9	18
ServicesViewModel	88	6	1	7	7
ServiceViewModel	90	7	1	6	11
EistCockpit.ServicesPlugin.Loader (Debug)	91	8	1	8	8
EistCockpit.Plugins.ServicesPlugin.View	91	8	1	8	8
HostPluginInitializer	91	8	1	8	8

Abbildung 82: Metrics ServicesPlugin



9.6.3.6 AppConfigPlugin



Hierarchy	Mainta...	Cyclo...	Depth ...	Class C...	Lines of C...
EistCockpit.AppConfigPlugin.Loader (Debug)	91	14	1	13	17
EistCockpit.Plugins.AppConfigPlugin.Loac	91	14	1	13	17
AppConfigPluginInitializer	91	14	1	13	17
EistCockpit.AppConfigPlugin.View (Debug)	100	0	0	0	0
EistCockpit.AppConfigPlugin.ViewModel (Deb	81	216	1	34	398
EistCockpit.Plugins.AppConfigPlugin.View	81	216	1	34	398
AppConfigViewModel	74	64	1	14	113
GeneralSettingsViewModel	82	13	1	10	24
ISettingsView	100	3	0	0	0
PluginAccessSettingsViewModel	64	69	1	25	144
RolesSettingsViewModel	72	62	1	25	110
Topic	93	5	1	1	7

Abbildung 83: Metrics AppConfigPlugin



9.6.4 Coding Conventions

Generell gelten die „Coding Conventions for C# 4.0“ [28]für das EistCockpit.

9.6.4.1 Formatierung

Für die Formatierung eines Sourcecode-Files verwenden wird die im Visual Studio verfügbare „Format Code“-Funktion (Ctrl+E,D). Dies beinhaltet das Einrücken, Zeilenlänge und die Position von Klammern.

9.6.4.2 Kommentare

Die Kommentare sind generell in Englisch zu halten. Kommentare mit Dokumentation-charakter sind im Doxygenformat einzupflegen, während einfache Kommentare möglichst kurz auf einer Zeile zu halten sind. Diese sind dann immer mit „“//“ zu machen.

---

[28] [hunt07]: C# Coding Standards for .NET Version 1.15
http://se.inf.ethz.ch/old/teaching/ss2007/251-0290-00/project/CSharpCodingStandards.pdf
(08.06.2012)



9.6.4.3 Namenskonventionen

Die Namenskonventionen sind in Anlehnung an diese Coding Conventions wie folgt zu beachten:

- „c“ camelCase
- „P“ PascalCase
- „_“ Präfix mit _Underscore
- „U“ UPPERCASE_UNDESCORE

Subjekt	Public	Protected	Internal	Private	Bemerkung
Projektdatei	P				Muss mit Namespace übereinstimmen
Sourcedatei	P				Muss mit Klasse übereinstimmen
Andere Datei	P				
Namespace	P				
Klasse	P	P	P	P	
Interface	P	P	P	P	Präfix mit grossem „I“
Generische Klasse	P	P	P	P	„T“ oder „K“ als Identifier
Methode	P	P	P	P	Verb oder Verb-Objekt
Property	P	P	P	P	Nicht mit Get/Set
Member	P	_c	_c	_c	
Konstante	U	U	U	U	
Enum	P	P	P	P	
Delegate	P	P	P	P	
Event	P	P	P	P	
Parameter				c	

Tabelle 65 Code Namenskonventionen

9.6.5 Angewandte Patterns

Siehe Abschnitt 7.3.4 Patterns



9.6.6 Verwendete Libraries

9.6.6.1 .Net Framework

Autor	Microsoft (http://www.microsoft.com/)
Version	4.0.0.0
Quelle	http://www.microsoft.com/germany/net/net-framework-4.aspx
Lizenz	EULA
Verwendungsorte	Gesamter EistCockpit-Namespace
Zweck	Das .net Framework ist das Fundament und die Bausubstanz des EistCockpit.

Tabelle 66 Library .NET Framework

9.6.6.2 SQLite

Autor	SQLite.org
Version	1.0.66.0
Quelle	http://system.data.sqlite.org/
Lizenz	Public Domain (http://sqlite.org/copyright.html)
Verwendungsorte	• EistCockpit.DAL • SQLite-Prototyp • ExampleDataGenerator
Zweck	Wie aus den Designeintscheidungen aus dem Dokument „Entwurf“ hervorgeht, wird für das EistCockpit SQLite als Datenbank verwendet. Um Diese jedoch zweckmässig einsetzen zu können bedarf es dieses Datenbanktreibers für das Entity Framework (EF).

Tabelle 67 Library SQLite

9.6.6.3 WPFToolkit.Extended

Autor	Xceed (http://www.codeplex.com/site/users/view/Xceed)
Version	1.6.0
Quelle	http://wpftoolkit.codeplex.com
Lizenz	Microsoft Public License (http://wpftoolkit.codeplex.com/license)
Verwendungsorte	• EistCockpit.Plugins.MessagesPlugin.View • EistCockpit.Plugins.LocationsPlugin.View • EistCockpit.Plugins.AppConfigPlugin.View
Zweck	Die Standart WPF-Controls von .net 4.0 enthalten zwar einen Datepicker, jedoch fehlt eine einfache Möglichkeit darüber auch die Zeit einzugeben. Das WPFToolkit enthält dieses und weitere nützliche Controls.

Tabelle 68 Library WPFToolkit.Extended



9.6.6.4 Microsoft Extensibility Framework (MEF)

Autor	Microsoft (http://www.microsoft.com/)
Version	4.0.0.0
Quelle	http://mef.codeplex.com
Lizenz	Microsoft Public License (http://mef.codeplex.com/license)
Verwendungsorte	• EistCockpit.Loader • EistCockpit.Plugins.MessagesPlugin.Loader • EistCockpit.Plugins.LocationsPlugin.Loader • EistCockpit.Plugins.AppConfigPlugin.Loader
Zweck	Das Microsoft Extensibility Framework vereinfacht den Prozess des Dynamischen ladens von DLLs, was ein wesentlicher Bestandteil unserer Plugin-Architektur darstellt.

Tabelle 69 Library MEF



9.6.7 Fehler & Warnungen

Aus den nachstehenden Abbildungen wird ersichtlich, dass keine Fehler bzw. Warnungen beim Kompilieren der Projekte erscheinen.

9.6.7.1 EistCockpit

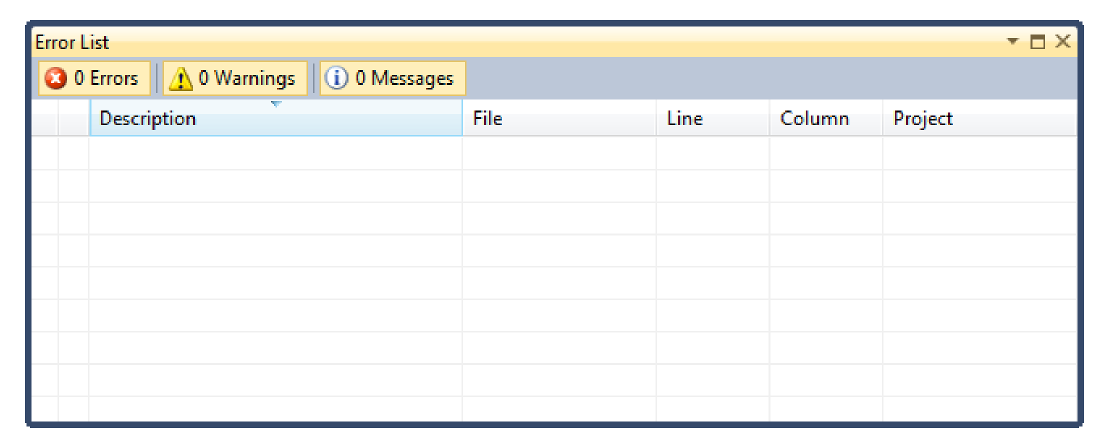


Abbildung 84: Errors und Warnings EistCockpit

9.6.7.2 EistCockpit.Plugins.MessagesPlugin

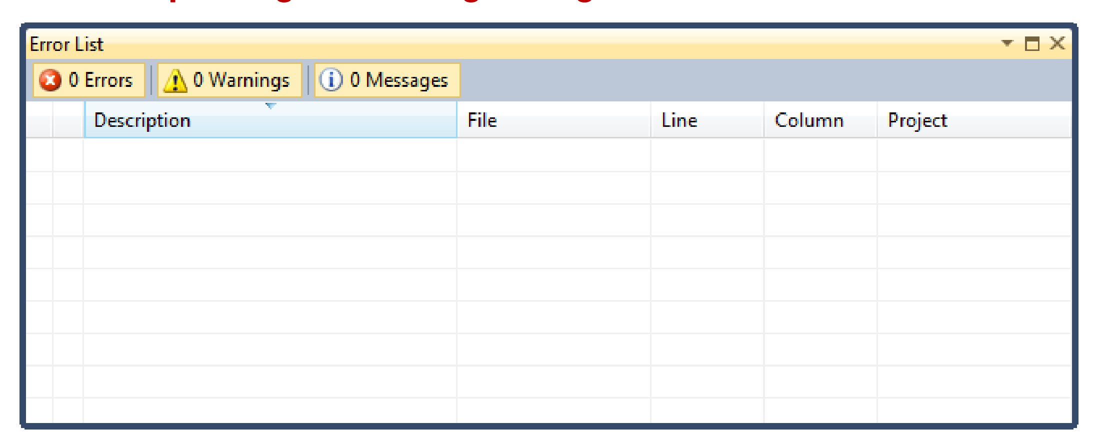


Abbildung 85: Errors und Warnings MessagesPlugin

9.6.7.3 EistCockpit.Plugins.LocationsPlugin

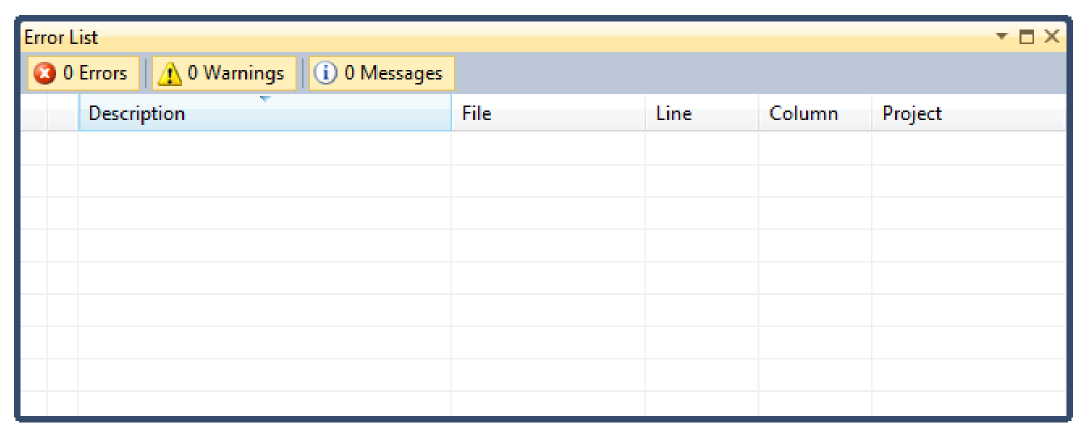


Abbildung 86: Errors und Warnings LocationsPlugin



9.7 Code Reviews

Wie im Qualitätsmanagement festgehalten, muss für dieses Projekt mindestens einmal ein Code Review mit dem Betreuer durchgeführt werden. Nachfolgend sind diese Dokumentiert.

9.7.1 Sprint 3

Datum	06.06.2012
Betreuer	Michael Gfeller
Stand	29.05.2012

Zusammenfassung

- Projekt muss auf einem anderen Entwicklungsrechner installiert werden können
- Logger fehlt
- Converter Klassen gehören nicht ins Common
- Namenskonventionen auch bei DTOs einhalten
- Enum [MessageType] ist wie ein Flag Enum nummeriert aber nicht als [Flag] definiert
- MessagePriority.Urgent wird im GUI nicht dargestellt
- Liste der Radiobuttons könnte generiert werden
- Lokalisierung eingeplant?
- 2 Singeltons wurden verwendet
- 1 Singelton ist meistens in Ordnung, ein ServiceLocator welcher beim Startup konfiguriert wird. z.B. http://unity.codeplex.com/
- Regionen /* --- IVARS --- */ in C# via #Region IVARS #Endregion
- Resize von GUI's alle auch bei „Full-Screen" anschauen.
- PluginSupport besser als Basisklasse oder ein ServiceLocator dem Plugin übergeben?
- Unit Tests benötigen File zum Laufen, könnte in den Test-Settings definiert werden
- INotifyPropertyChanged wird an verschiedenen Orten implementiert
- Resourcen könnten als Applikationsresourcen definiert werden
- Namespace wird beim Location-Plugin nicht einheitlich implementiert

Beschlüsse

Da das Code Review sehr spat stattgefunden hat, werden die beanstandeten Codefragmente als Teil der Bachelorarbeit überarbeitet. Kleinere Korrekturen können jedoch schon vorher eingepflegt werden:

- Entwicklerdokumentation soll eine Anleitung für andere Entwickler bieten
- Enum MessageType wurde bereits mit dem Flag versehen
- MessagePriority.Urgent soll dem System vorenthalten bleiben
- Lokalisierung ist sowieso ein Teil der Bachelorarbeit
- Singeltons könnten mit Unity abgelöst werden
- Kommentare und Regions wurden bereits überarbeitet
- Namespace im Locationplugin wird noch vereinheitlicht



9.8 Externes Design

Dieser Abschnitt gibt einen kurzen Überblick über das schlussendlich realisierte externe Design.

9.8.1 Plugin Standorte

9.8.1.1 Standort anzeigen

Die folgende Abbildung (Verweis Abbildung 87: Externes Design Locations) zeigt die Übersicht des Standort Plugins, bei der zu einem ausgewählten Standort die dazugehörigen Informationen angezeigt werden.

Abbildung 87: Externes Design Locations



9.8.1.2 Standort bearbeiten

Dieses modale Fenster (siehe Abbildung 88: Externes Design Location Detail) wird aufgerufen, sobald man auf die Schaltfläche Bearbeiten drückt. Es dient dazu einen bereits erfassten Standort zu bearbeiten.

Standort erfassen

Standort Standort 3

Allgemein

Kurzbezeichnung*	Standort 3
Bezeichnung*	Bezeichnung Stao 3
Beschreibung	Eine Beschreibung zu einem Standort
Chef*	Hans Meier - Telefonnr. 0449328328
Einheit*	RistlBat 20

Lage

Koordinaten*	500000/100000
Adresse*	Musteradresse 3783

Landbesitzer

Name Landbesitzer*	Hans Müller
Adresse Landbesitzer*	Müllerstr. 382 3838 Müllhausen
Telefonr. Landbesitzer*	89382983

Belegt

Von*	Donnerstag, 7. Juni 2012 15:54:01
Bis*	Donnerstag, 7. Juni 2012 15:54:02

Speichern Abbrechen

Abbildung 88: Externes Design Location Detail



9.8.1.3 Standort erfassen

Beim Erfassen eines neuen Standorts wird dasselbe modale Fenster aufgerufen wie beim Bearbeiten. Im untenstehenden Screenshot (siehe Abbildung 89: Externes Design Location erfassen) sieht man zudem noch Felder die vom Benutzer falsch ausgefüllt wurden, folge dessen schlägt die Validierung fehl dies bewirkt die Anzeige eines Hinweises.

Standort erfassen

Standort

Allgemein

Kurzbezeichnung*

Es muss eine Kurzbezeichnung für den Standort angegeben werden.

Bezeichnung*

Beschreibung

Chef*

Einheit*

Lage

Koordinaten* 000000/000000

Der Standort muss in der Schweiz liegen

Adresse*

Landbesitzer

Name Landbesitzer*

Adresse Landbesitzer*

Telefonr. Landbesitzer*

Belegt

Von* Donnerstag, 7. Juni 2012 15:56:36

Bis* Donnerstag, 7. Juni 2012 15:56:37

Speichern | Abbrechen

=>Felder nicht korrekt
=>Koordinaten nicht korrekt
=>Keine Einheit ausgewählt
=>Keine verantwortliche Person ausgewählt

Abbildung 89: Externes Design Location erfassen



9.8.2 Plugin Meldungen

9.8.2.1 Meldungen anzeigen/filtern

Beim Plugin Meldungen handelt es sich um das Kernplugin, welches die Funktionalitäten enthält, welche am dringendsten benötigt werden. In der Übersicht (siehe Abbildung 90: Externes Design Meldung Übersicht) werden sämtliche erfassten Meldungen angezeigt sowie Filtermöglichkeiten angeboten um nach bestimmten Kriterien zu filtern.

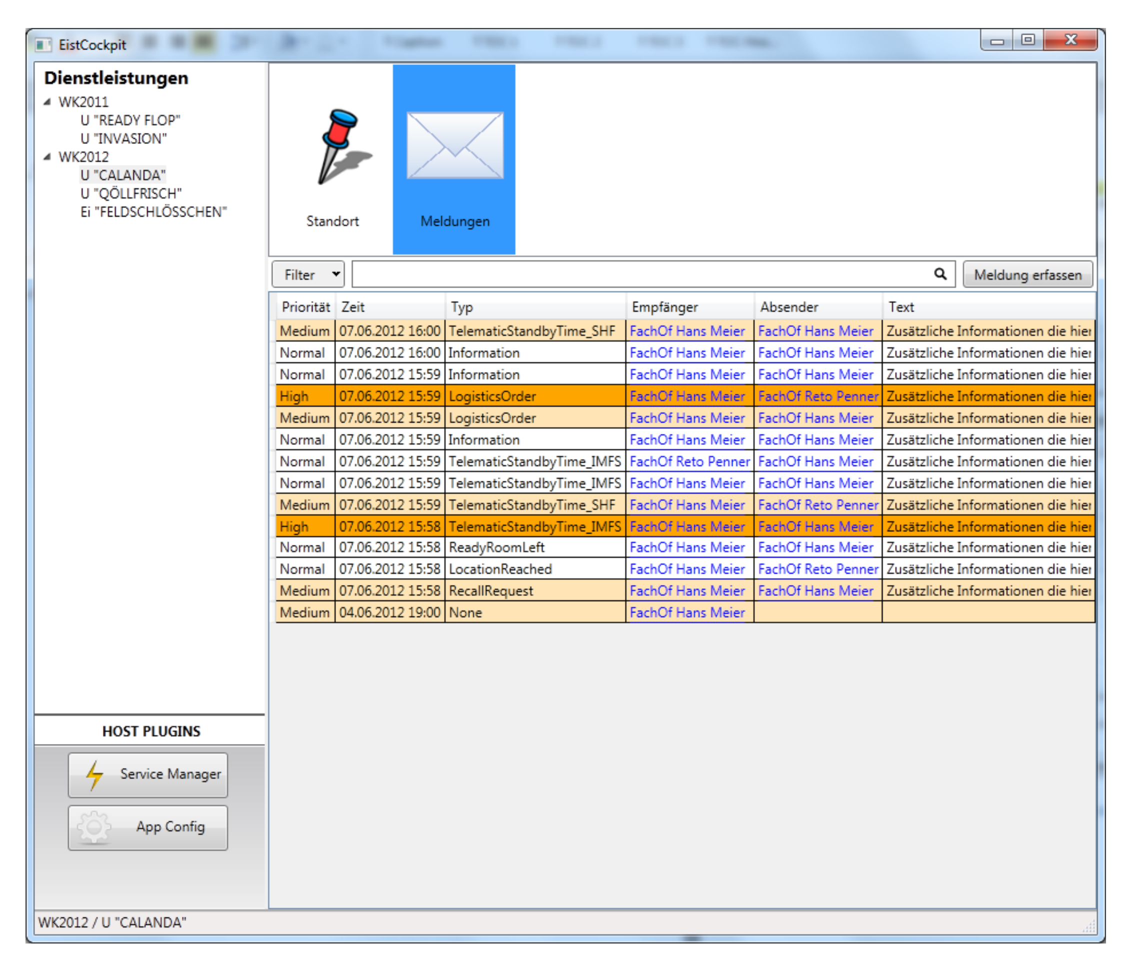


Abbildung 90: Externes Design Meldung Übersicht



9.8.2.2 Meldungen erfassen

Die Abbildung 91: Externes Design Meldung erfassen gibt einen Überblick über das Formular zum Erfassen von neuen Meldungen. Dabei wurde auf eine gute Benutzerführung mittels unterstützenden Beschriftungen geachtet.

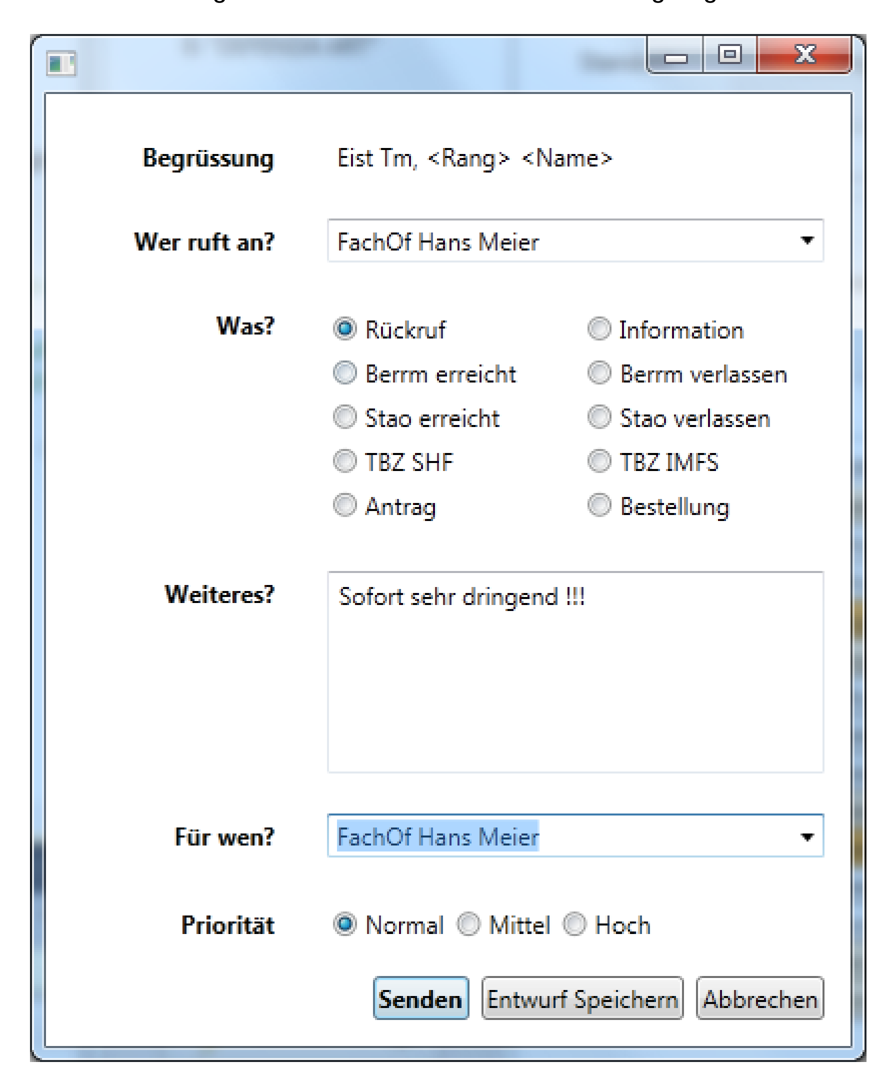


Abbildung 91: Externes Design Meldung erfassen



9.8.3 Service Manager

Mittels des Service Managers können bestehende Operationen, Dienstleistungen bearbeitet sowie neue Operationen und Dienstleistungen erfasst werden.

9.8.3.1 Operationen bearbeiten

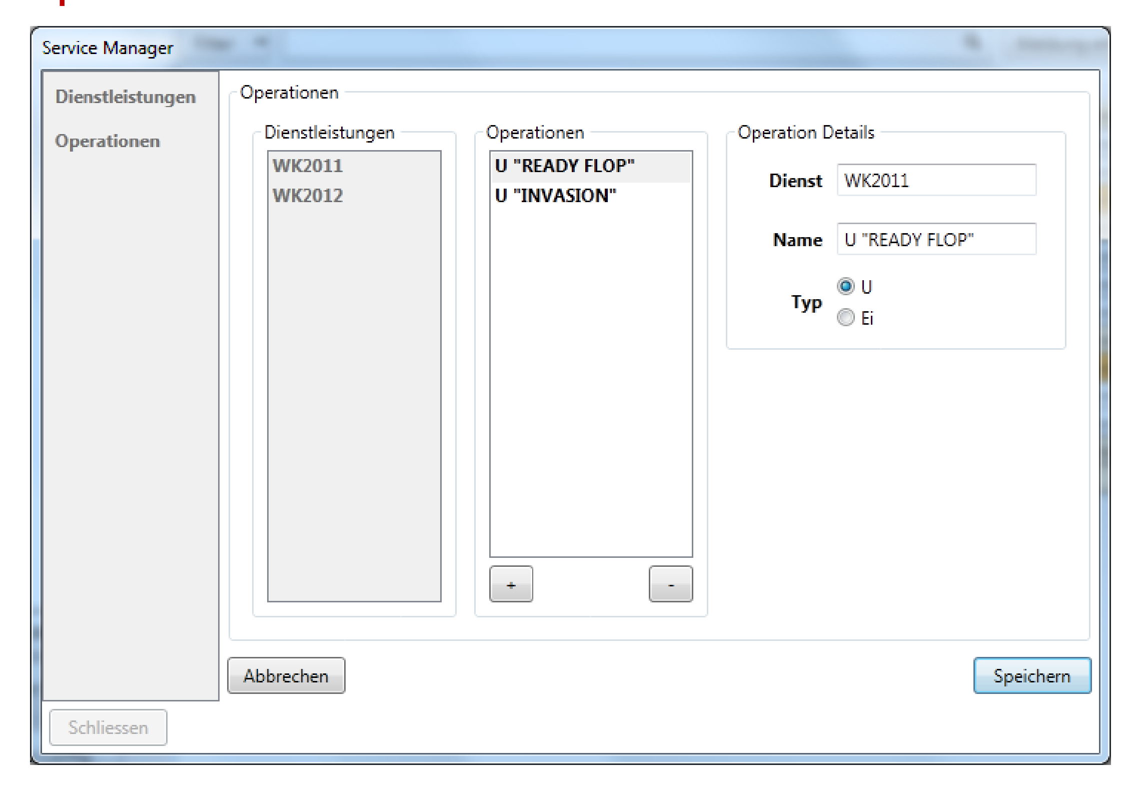


Abbildung 92: Externes Design Operation bearbeiten

9.8.3.2 Dienstleistungen bearbeiten

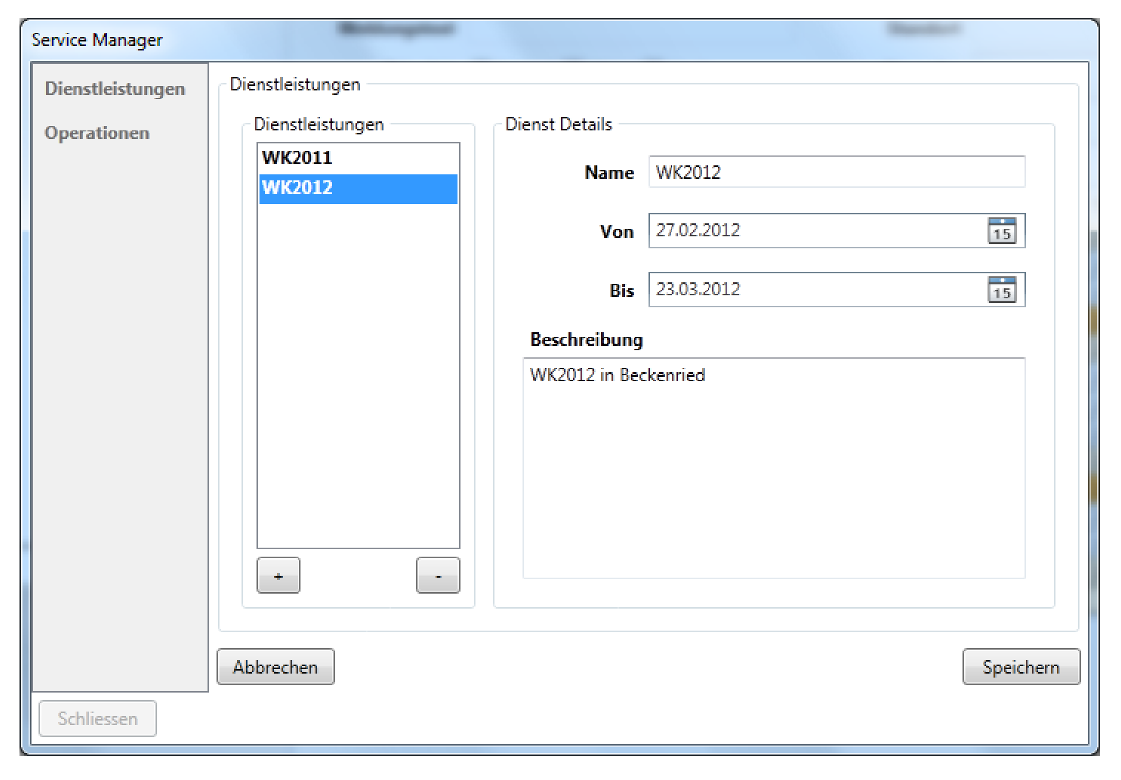


Abbildung 93: Externes Design Dienstleistung bearbeiten



9.8.4 App Config

Das App Config enthält alle Einstellungen, welche für die gesamte Applikation relevant sind.

9.8.4.1 Allgemein

Unter dem Register Allgemein (siehe Abbildung 94: Polling Intervall) kann der Polling Interval mithilfe eines Reglers definiert werden.

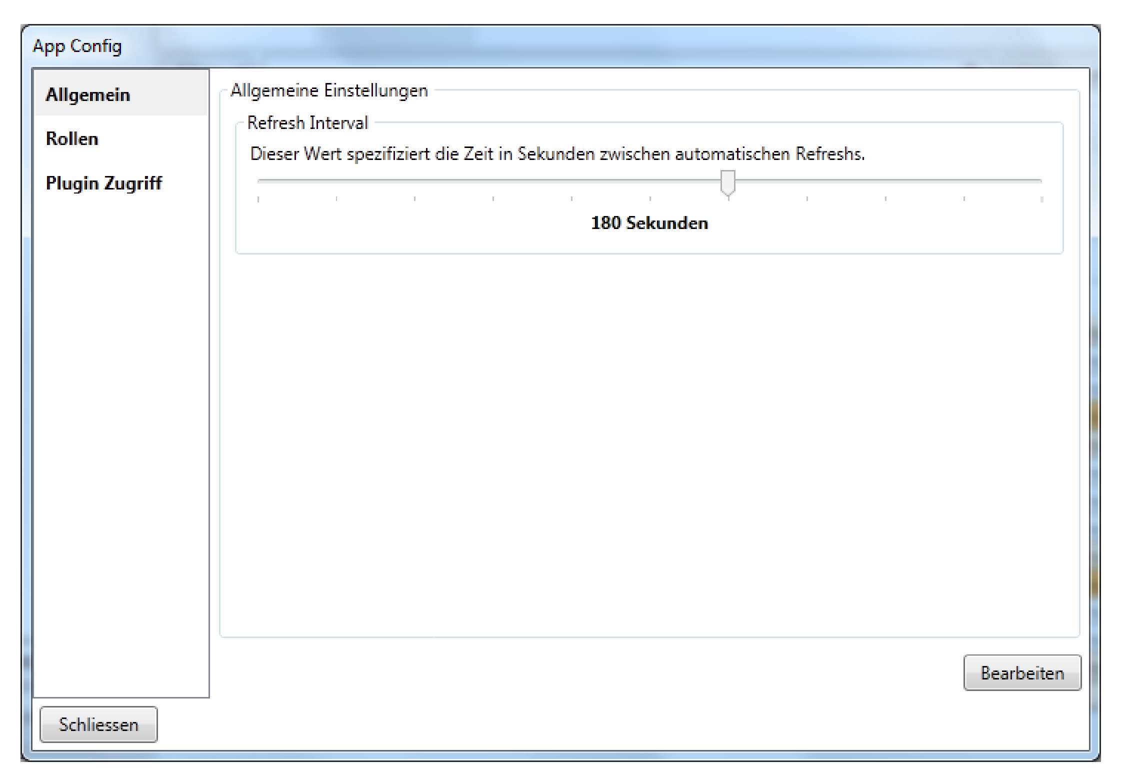


Abbildung 94: Polling Intervall



9.8.4.2 Rollen

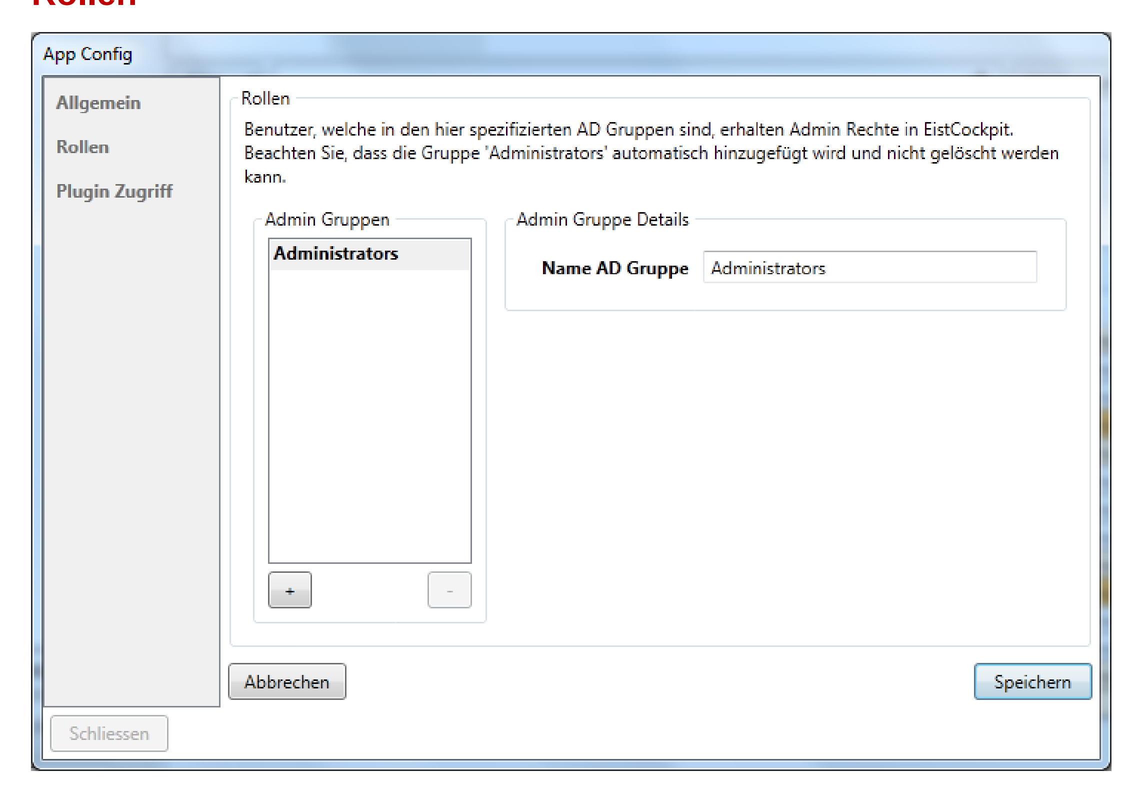


Abbildung 95: AD-Gruppen

9.8.4.3 Plugin Zugriff

Das Register „Plugin Zugriff" enthält eine Auflistung sämtlicher geladener Plugins mitsamt nützlichen Informationen wie Name, Version, Typ, Beschreibung etc.

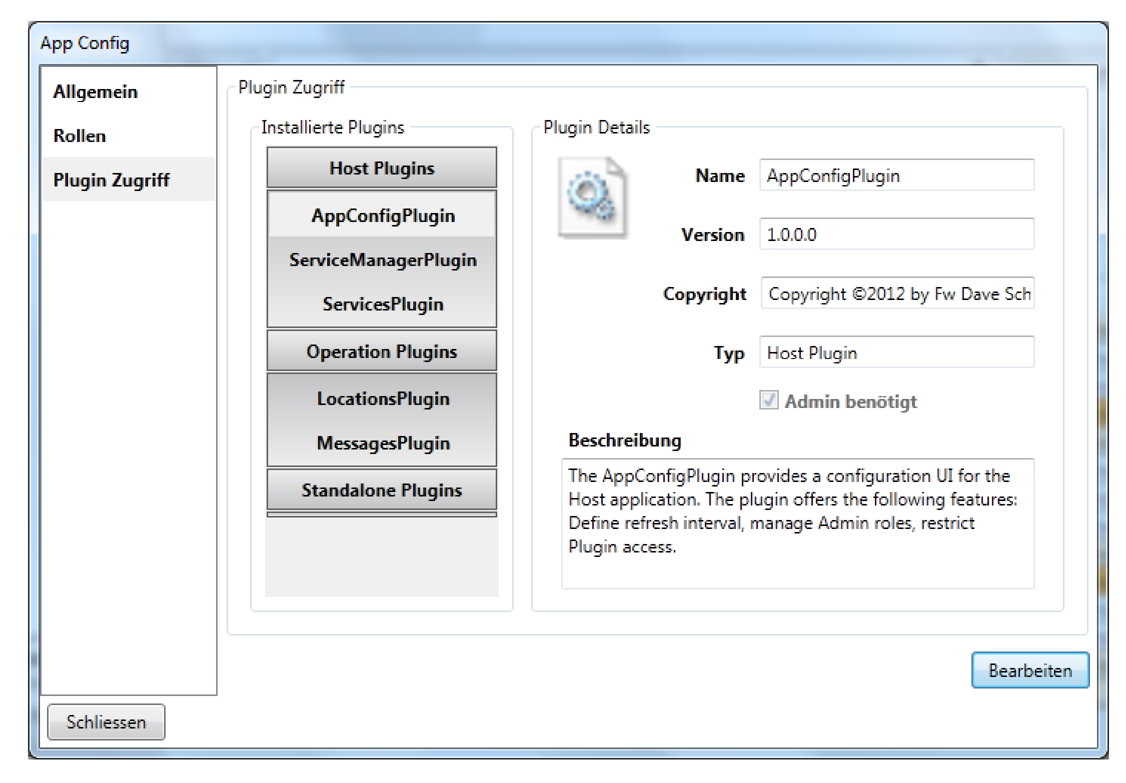


Abbildung 96: Plugin Access



EistCockpit

10 Developer Manual

David Schöttl, Remo Waltenspül & Diego Steiner



10.1 Dokumenteninformation

Datum	Version	Änderung	Autor
07.06.2012	1.0	Erste Version des Dokuments	rwaltens
07.06.2012	1.1	Kleinere Fehler behoben	dsteiner
08.06.2012	1.2	Review	rwaltens



10.2 Übersicht

Das vorliegende Dokument beschreibt wichtige Informationen und gibt Hinweise, welche für die Weiterentwicklung von EistCockpit bekannt sein müssen.

Es wird davon ausgegangen, dass der Leser ein Softwareentwickler ist, welcher mit den verwendeten Technologien (.NET, WCF, WPF) vertraut ist.

10.2.1 Komponenten

Nachfolgend werden die Komponenten von EistCockpit beschrieben; dabei wird zwischen Server-Komponenten und Client-Komponenten unterschieden.

10.2.1.1 Serverseitig

Serverseitig besteht EistCockpit aus folgenden Komponenten:

- **Datenbank** – SQLite Datenbank für die Persistenz.
- **DAL** – Data Access Layer.
- **Webservice** – WCF-basierter REST/JSON Service.

10.2.1.2 Clientseitig

Clientseitig besteht EistCockpit aus folgenden Komponenten:

- **Host Applikation** – Die Host Applikation stellt das UI Grundgerüst zur Verfügung, welches von den eingebundenen Plugins genutzt wird, um Inhalte darzustellen.
- **Plugins** – Verschiedene Plugins, welche von der Host Applikation eingebunden werden und spezifische Funktionalität/Features, einschliesslich UI, implementieren. Es wird zwischen den Typen 'Host Plugin', 'Operation Plugin' und 'Standalone Plugin' unterschieden.[29]

10.2.2 Voraussetzungen

Nachfolgend werden die Systemvoraussetzungen beschrieben, welche für die Entwicklung sowie Ausführung von EistCockpit erfüllt sein müssen:

- **.NET Framework 4.0** – Das .NET Framework 4.0 muss server- wie auch clientseitig installiert sein.
- **IDE** – EistCockpit wird mit der aktuellen Version von Microsoft VisualStudio[30] entwickelt.
- **Project Properties** – Es muss darauf geachtet werden, dass bei neu erstellten Projekten explizit gegen '.NET Framework 4.0' gelinkt wird; bei WPF Projekten wird per Default gegen '.NET Framework 4.0 (Client)' gelinkt, was unzureichend ist.
- **ADO.NET Adapter** – Da eine SQLite DB zum Einsatz kommt, muss ein entsprechender ADO.NET Adapter installiert werden; EistCockpit verwendet den Adapter 'System.Data.SQLite'.[31]

---

[29] Für genauere Informationen zu den verschiedenen Plugin Typen wird auf das Entwurfs-Dokument verwiesen.
[30] Zum Zeitpunkt der Verfassung dieses Dokuments: VisualStudio 2010
[31] [url18]: System.Data.SQLite: About, http://system.data.sqlite.org/index.html/doc/trunk/www/index.wiki (04.06.2012)



10.3 Entwicklung

10.3.1 Allgemeines

EistCockpit verwendet mehrere Solutions mit entsprechenden Unterprojekten.

Die Serverkomponente sowie die Host Applikation werden in der sog. Core Solution 'EistCockpit.sln' (auf erster Ebene des Source Directories) entwickelt, während für die Plugins jeweils einzelne Solutions verwendet werden (Beispiel App Config Plugin: 'EistCockpit.Plugins.AppConfigPlugin.sln').

Solution Directories für die einzelnen Plugins werden im Directory 'EistCockpit.Plugins' (auf erster Ebene des Source Directories) abgelegt.

10.3.2 Coding Guidelines

Die Form des Source Codes sollte sich an den offiziellen C# Coding Conventions[32] orientieren, wobei sinnvolle Abweichungen davon durchaus legitim sind.[33]

Die einzelnen Teile einer Klasse sollten durch den Einsatz von *#region* Konstrukten strukturiert werden; es ist dem Entwickler jedoch ausdrücklich freigestellt, ob er *zusätzliche* Kommentar-Konstrukte zur Strukturierung verwenden will.

Die Source Code Dokumentation wird mit Doxygen[34] generiert, daher sollten *Public* Members sowie Klassen und Namespaces unter der Verwendung der entsprechenden Doxygen Tags[35][36] dokumentiert werden.

Innerhalb des Source Codes, einschliesslich Dokumentation und Kommentare, kommt die englische Sprache zum Einsatz.

10.3.3 Core Solution

Nachfolgend wird auf einige Aspekte eingegangen, welche bei der Entwicklung innerhalb der Core Solution beachtet werden müssen.

10.3.3.1 Datenbank

SQLlite Datenbanken bestehen aus einem einzigen File; EistCockpit verwendet den Namen 'EistCockpitDB.db'.

Die Master DB befindet sich im Directory 'DataBase' (auf erster Ebene des Source Directories) – dies ist die Master DB, was bedeutet, dass diese *NICHT* von der ausgeführten EistCockpit Instanz verwendet werden sollte!

Die Master DB wird für den Betrieb an einen definierten Ort kopiert. Der Pfad zu dieser Working DB wird im App.config (bzw. Web.config) File des entsprechenden Projekts,

---

[32] [url20]:C# Coding Conventions, http://msdn.microsoft.com/en-us/library/ff926074.aspx (05.06.2012)
[33] Beispiel: Privat Members beginnen mit einem Lower Case Buchstaben, während Public Members mit einem Capital Case Buchstaben beginnen.
[34] [url19]: Doxygen, http://www.doxygen.org (06.06.2012)
[35] [url21]: Documenting the Code, http://www.stack.nl/~dimitri/doxygen/docblocks.html (06.06.2012)
[36] [url22]: Commands, http://www.stack.nl/~dimitri/doxygen/commands.html (06.06.2012)



welches mit der DB arbeiten will, definiert. Da diese Config-Files im Repository eingecheckt sind und mehrere Mitarbeiter auf verschiedenen Maschinen an der Entwicklung beteiligt sind, gilt es, einen Pfad zu definieren, welcher auf allen Entwicklungs-Maschinen vorhanden ist.

Der **Webservice** verwendet folgenden Pfad für die Working DB:

- C:\ProgramData\EistCockpit\EistCockpitDB.db

Das **UnitTest Projekt** für den DAL verwendet folgenden Pfad für die DB:

- C:\ProgramData\EistCockpit\EistCockpitDB_Test.db

Das **ExampleDataGenerator**[37] Projekt verwendet folgenden Pfad für die DB:

- C:\ProgramData\EistCockpit\EistCockpitDB_ExampleData.db

Es liegt in der Verantwortung des Entwicklers, sicherzustellen, dass die Pfade in den Config-Files gültig sind.

10.3.3.2 DAL

Der Data Access Layer (EistCockpit.DAL) ermöglicht den Zugriff auf die Persistenzschicht und bietet dazu grundlegende CRUD Operationen für die verschiedenen Entities an.

Der Zugriff auf die DAL Funktionalität sollte nie direkt erfolgen, sondern über den Business Layer (EistCockpit.Business), welcher seinerseits die Anfragen an den DAL weiterleitet. Grund dafür ist, dass damit die auditierbarkeit des Systems erhalten bleibt – falls dies künftig eine Anforderung an EistCockpit wird.

Die verwendeten Entity-Klassen wurden aus dem Datenmodell generiert; dieses wiederum basiert auf der erwähnten Master DB.

Bezüglich 'Entites' muss erwähnt werden, dass *zwei* Arten von Entities in EistCockpit verwendet werden:

- **Generierte Entities (Entity)** – Die aus dem Datenmodell generierten Entitäten werden bis- und mit des serverseitigen Teils des Webservice verwendet; sie dienen als DTOs im Rahmen der Datenbank.
- **Re-Implementationen (DTO)** – Re-Implementationen der generierten Entities (siehe Namespace 'EistCockpit.Common.DTOs', Naming Schema '<Entity>DTO.cs') werden vom Webservice für die Kommunikation mit den Clients verwendet. Grund dafür ist, dass die generierten Entitäten vom Framework nicht korrekt in JSON Objekte serialisiert werden können.[38]

Obwohl die Re-Implementation der Entities und die Konversion zwischen den beiden Typen einen gewissen Grad an Redundanz und Aufwand bedeutet, hat dies auch den Vorteil, dass unnötiger Ballast, welcher in den generierten Entities vorhanden ist, nicht an die Clients geschickt werden muss. Die DTOs müssen lediglich die direkten Properties der Entities implementieren, denn die Navigations-Properties werden über die REST API, d.h.

---

[37] ExampleDataGenerator Projekt; dient zur Generierung von Beispieldaten: Dirty Hack Entwicklungs-Artefakt, welches kein Deliverable von EistCockpit darstellt.
[38] Die genaue Ursache für dieses Verhalten wurde nicht genauer untersucht. Es wird jedoch vermutet, dass die generierten Annotations (insbesondere 'Reference') dafür verantwortlich sind.



den Webservice, realisiert. Weitere Vorteile der DTOs sind die Verwendbarkeit von Enums sowie die klare Trennung zwischen Server- und Clientseite.

Bei der Entwicklung sollte deshalb darauf geachtet werden, dass serverseitig nur mit den generierten Entities gearbeitet wird, während clientseitig nur die DTOs eingesetzt werden. Die Konversion zwischen Entity und DTO geschieht jeweils im Webservice.

10.3.3.3 Webservice

Der EistCockpit Webservice verwendet REST als API und JSON als Datenformat. Wie unter WCF üblich besteht der Service aus einer Service Implementation (siehe Namespace 'EistCockpit.Webservice', Klasse 'EistCockpitService') und einer Implementation für den Client Proxy (siehe Namespace 'EistCockpit.Common', Klasse 'EistCockpitServiceClient'), welche beide einem Service Interface entsprechen (siehe Namespace 'EistCockpit.Common', Interface 'IeistCockpitService').

Die Service Implementation greift via Business Layer (siehe Namespace 'EistCockpit.Business', Klasse 'EistCockpitBusinessLayer') auf die grundlegenden CRUD Operationen des DAL zu, um so die entsprechenden REST API Methoden zu implementieren. Wie bereits erwähnt ist die Service Implementation für die Konvertierung zwischen Entities und DTOs zuständig.

Die Client Proxy Implementation verwendet Channeling; dabei wird der Aufruf lediglich an die Basis-Klasse weitergeleitet. Dies ist möglich, weil es sich um einen .NET Client handelt.

10.3.3.4 Host Applikation & Loader

Die Host Applikation stellt lediglich ein Grundgerüst für die Plugins dar; das UI besteht aus einem weitgehend leeren Window.

Das Loader Projekt wird als Startup Project für die Host Applikation verwendet. In der aktuellen Entwicklungs-Version wird davon ausgegangen, dass im Project Directory 'EistCockpit.Loader' (auf erster Ebene des Source Directories) ein Directory 'Plugins' existiert. Aus diesem Plugin Directory werden die Plugins geladen, weshalb alle Plugin Projekte dieses Plugin Directory für den Build Output verwenden sollten.

Das Einfügen der Plugin UIs in das UI der Host Applikation geschieht im Main View Model (siehe Namespace 'EistCockpit.ViewModels', Klasse 'MainViewModel').

10.3.4 Plugin Solutions

Nachfolgend wird auf einige Aspekte eingegangen, welche bei der Entwicklung von Plugins beachtet werden müssen.

10.3.4.1 Genereller Aufbau

Plugin Solutions enthalten jeweils drei Class Library Subprojekte:

- **Loader** – Enthält eine Initializer Klasse, welche das entsprechende Plugin Interface implementiert.
- **Model** – Enthält View Models für die einzelnen Views.
- **View** – Implementiert das UI.

Es sollte darauf geachtet werden, dass die Namespace Konventionen (EistCockpit.Plugins.<PluginName>.<Subproject>) eingehalten werden.



10.3.4.2 Project Settings

Auch für Plugins gilt die Vorgabe, dass gegen das '.NET Framework 4.0' gelinkt werden muss; dies ist WPF Projekten ggf. zu korrigieren.

Die DLL des *View* Projekts wird schlussendlich als Plugin verwendet, deshalb müssen in den Project Settings folgende Einstellungen vorgenommen werden:

- **Assembly Information** – Die Assembly Information wird vom AppConfigPlugin ausgelesen und dargestellt. Benötigte Angaben (in *Englisch*[39]): Title, Version, Copyright, Description.
- **Build Output** – Wie bereits erwähnt sucht die Host Applikation die Plugins im 'Plugin' Directory innerhalb des 'EistCockpit.Loader' Directories. Daher sollte der Build Output mit einer relativen Pfadangabe auf dieses Directory umgelenkt werden.

Für die anderen Subprojekte ist die Angabe der Assembly Informationen optional, jedoch wünschenswert. Der Build Output muss bei diesen Projekten nicht umgeleitet werden.

[39] Die Attribute sollen in Englisch angegeben werden, weil es sich dabei um Informationen handelt, welche ausschliesslich für Entwickler/Administratoren von belang sind – und nicht für den normalen Benutzer. Da die Entwicklungssprache Englisch ist, sind auch diese Angaben in Englisch zu halten.



EistCockpit

11 Schlussfolgerung

David Schöttl, Remo Waltenspül & Diego Steiner



11.1 Dokumenteninformation

Datum	Version	Änderung	Autor
07.06.2012	1.0	Erste Version des Dokuments	rwaltens
07.06.2012	1.1	Ergebnisse beschrieben	dsteiner
07.06.2012	1.2	Ausblick dokumentiert	rwaltens



11.2 Einleitung

11.2.1 Ergebnisse

Ziel war es, mit der Studienarbeit ein verteiltes Informations-/Status-/Meldesystem zu entwickeln, dass im Stab eingesetzt werden kann um die Mängel in den vorhandenen Arbeitsabläufen zu Eliminieren.

- **Anforderungen aufnehmen**
  Zu Beginn der Studienarbeit besuchten alle Projektmitglieder den Wiederholungskurs beim Ristl Bat 20 (Kunde). Dort konnten alle Anforderungen aufgenommen werden und die Entwickler bekamen ein umfassendes Bild über die Arbeitsabläufe und die Umgebung. Diese wurden genau dokumentiert und dienen als Basis sowohl für die Studienarbeit als auch für die allfällige Bachelorarbeit, die daraus hervorgehen soll.
- **Basis für Applikation schaffen**
  Damit die angestrebten Applikationsteile einfach auf dem System installiert werden können, wurde mit der HostApplikation des EistCockpits eine solides Fundament für Plugins geschaffen. So wurde das Ziel erreicht, dass unabhängige Applikationsteile ohne grossen Aufwand in das System eingeklinkt werden können. Dies kommt auch der Sicherheit zugute, weil neue Versionen der Plugins jeweils nicht durch einen Mitarbeiter der RUAG installiert werden müssen, sondern bequem vom Personal vor Ort eingerichtet werden können.
- **Elektronischen Meldezettel umsetzen**
  Das Problem der dem Projekt den Anstoss gegeben hat war die ineffizienten Papiermeldezettel durch ein effizienteres, Elektronisches System zu ersetzen. Mit dem Meldungsplugin wurde der Prototyp für diesen Zweck geschaffen. Obwohl es grundsätzlich bereits funktioniert, ist es noch nicht einsatzreif; dafür bedarf es neben weiteren Kleinigkeiten des Personen-Selektors, mit welchem sich Personen als Absender oder Empfänger schnell auswählen lassen sollen.
- **Informationsübersicht bieten**
  Ein weiteres Ziel war es, dem Benutzer eine zentrale Anlaufstelle für Informationen rund um die Eist anzubieten. Mit dem Standorteplugin ist ein Teil dessen bereits umgesetzt. Für eine vollständige Abdeckung fehlen aber unter anderem noch die Telematikbereitschaftszeiten.
- **Installation & Benutzerdokumentation**
  Damit das System eingesetzt werden kann, wird eine einfache Installation und eine umfassende Benutzerdokumentation benötigt. Weil das Produkt aber noch nicht die Einsatzreife erreicht hat, muss im Rahmen der Studienarbeit auf diese Punkte verzichtet werden.



11.2.2 Vergleich zur bestehenden Lösung

Wie aus der Vorstudie hervorgeht, wird zurzeit keine Software eingesetzt um Meldungen zu verwalten. Ein Einsatz der Anwendung EistCockpit bietet nachstehende Vorteile:

- Meldungen können beinahe ohne Zeitverlust an einen bestimmten Empfänger übermittelt werden.
- Die Übersicht über bereits erfasste Meldungen ist bei weitem grösser.
- Die Mitarbeiter in der Triage werden durch Hilfetexte beim Erfassen von Meldungen unterstützt.
- Alle relevanten Informationen zu einem Standort werden übersichtlich dargestellt.
- Mittels Filtern können Meldungen nach bestimmten Kriterien angezeigt werden.
- Meldungen können anhand einer Priorisierung nach Wichtigkeit unterschieden werden, damit kann der Empfänger wichtigere von weniger wichtigen Meldungen auseinander halten.
- Die Erfassung von Meldungen über das elektronische Formular ist für einen AdA der nur eine Hand frei hat einfacher durchzuführen.



11.2.3 Ausblick

Die Applikation EistCockpit kann gegenwärtig noch nicht produktiv während dem normalen Dienstbetrieb eingesetzt werden. Um dies zu ermöglichen müsste man folgende Anforderungen in einer Folgearbeit umsetzen.

11.2.3.1 Allgemein

- Um die Benutzbarkeit und die geforderten nicht funktionalen Anforderungen zu verifizieren sind Usability Tests notwendig.
- Zurzeit existiert noch keine Bedienungsanleitung, diese muss für ein produktives System zwingend erstellt werden.
- Für eine Installation der Anwendung auf dem Server wird eine Installationsanleitung benötigt, die klar Schritt für Schritt definiert wie das Vorgehen ist um die Applikation zu installieren.
- Eine benutzergeführte automatische Installation (installer) wäre praktisch und würde die Einrichtung massgeblich erleichtern.
- Die Benutzeroberfläche sollte einheitlicher gestaltet werden.
- Für das Aufrufen von bestimmten Funktionen (Commands) sollen Shortkeys verwendet werden.
- Falls es zu einem Stromunterbruch kommt oder das System nicht mehr erreichbar ist, müssen die erfassten Meldungen regelmässig ausgedruckt werden.

11.2.3.2 Plugin Meldungen

- Die Kombobox bei der Auswahl des Anrufers/Empfängers müsste die Möglichkeit bieten nach Personen zu suchen (Personen Selektor)
- Es sollte möglich sein mehrere Empfänger beim Erfassen von Meldungen anzugeben.
- Beim Erfassen von Meldungen sollte mittels einer Validierung geprüft werden, dass eine Meldung nur versendet werden kann, wenn sämtliche notwendigen Felder ausgefüllt wurden.
- Es soll möglich sein eine Meldungshistory anzuzeigen, und daraus zu erkennen, wer zuletzt diesen Meldungstyp entgegengenommen hat.
- Die Erreichbarkeit von Zellenchefs soll im System gespeichert werden. Dadurch könnte eine Meldung an einen Stellvertreter weitergeleitet werden, falls der primäre Empfänger nicht erreichbar ist.
- Der Empfänger sollte aufgrund des gewählten Meldungstyps automatisch ausgewählt werden.
- Die Meldungen in der Übersicht sollten anhand des Typs gruppiert werden können.
- Eine Meldung sollte als erledigt markiert werden können.

11.2.3.3 Plugin Standorte

- Die Information zu einem Standort sollte eine Übersicht über vorhandenes Material bzw. Fahrzeuge enthalten (Disponibilität)
- Gegenwärtig werden zu einem Standort nur die Koordinaten angezeigt, anstelle dieser Koordinaten sollte man direkt eine Karte anzeigen, auf der der Standort eingetragen werden kann.
- Bei der Eingabe von Standortinformationen sollte dem Benutzer das erwartete Format angezeigt werden.



11.2.3.4 Plugin TBZ

Dieses Plugin soll die Erfassung und übersichtliche Darstellung der verschiedenen TBZ (Telematik-Bereitschafts-Zeiten) für SHF (Super High Frequency, Funk) bzw. IMFS (Integriertes militärisches Fernmelde-System) für die einzelnen Stao ermöglichen.

Funktionalität

- Darstellung der TBZ (erreicht/nicht erreicht, SOLL/IST)
- Bericht: Ausdruck der TBZ Tabellen in Plakat-Grösse

11.2.3.5 Plugin Netzplan

Dieses Plugin soll die grafische Darstellung eines IMFS Netzwerks ermöglichen, wobei Stao Infos, Meldungen, Phasen und dergleichen eingeblendet werden können. Die Informationen werden aus den Datenbeständen der anderen Plugins konsolidiert.

Funktionalität

- Darstellung des IMFS Netzwerks mit korrekten Symbolen für die verschiedenen Standorte (z.B. Vm, Rel, Kp Vm, Kl Vm mob etc.)
- Einblenden von verschiedenen Informationen (z.B. Stao Infos, Meldungen)
- Ansicht der Phasen
- Problem Indikatoren
- Verbindungsqualität
- Bericht: Ausdruck des Netzplans in Plakt-Grösse

11.2.3.6 Plugin Probleme & Lösungen

Dieses Plugin soll die Erfassung von neuem Wissen, Problemen und ähnlichem ermöglichen, sog. Lehren aus dem Krieg. Dies soll dabei helfen, Feststellungen zu erfassen, neue SOPs (Standard Operation Procedures) zu erstellen und Verbesserungsvorschläge zu sammeln, welche in künftigen Dienstleistungen von Vorteil sein können bzw. behandelt werden soll(t)en.

Funktionalität

- Erfassen von Inhalten
- Unterscheidung zwischen Problemen, Lösungen, Verbesserungsvorschlägen, etc.
- Stufengerechte Zugriffs-Beschränkungen

11.2.3.7 Plugin Wiki

Dieses Plugin soll die Erfassung von allgemeinen Informationen in Form eines Wikis ermöglichen, z.B. Wetterlage, News Flash, organisatorische Infos, etc.

Funktionalität

- Erfassen von neuen Inhalten
- Editieren der Inhalte
- Formatierung der Darstellung
- Stufengerechte Zugriffs-Beschränkungen



11.2.3.8 Plugin Vereinsverwaltung

Dieses Plugin soll die Verwaltung des Vereins "Freunde des Ristl Bat 20" vereinfachen.

Funktionalität

- Mitglieder-Verwaltung
- Finanz-Verwaltung, z.B. für Kiosk, Mitgliederbeiträge, Vereinsausgaben, etc.
- Termin-Verwaltung
- Dokument-Sammlung, z.B. Statuten, etc.
- Protokollierung, z.B. von Generalversammlungen, Beschlüssen, etc.
- Export-Funktion zur Übertragung der Informationen auf die Website des Vereins nach Dienstleistungsende

11.2.3.9 Plugin Gefechtsjournal

Dieses Plugin soll das physische Gefechts-Journal digitalisieren.

Funktionalität

- Gefechts-Journal Einträge erfassen
- Schlichte 1:1 Umsetzung des Gefechts-Journal in digitaler Form



EistCockpit

V. Projekt Retrospektive

David Schöttl, Remo Waltenspül & Diego Steiner



11.3 Dokumenteninformation

Datum	Version	Änderung	Autor
07.06.2012	1.0	Erste Version des Dokuments	dsteiner
07.06.2012	1.1	Auswertung Arbeitszeiten mit Diagrammen	dsteiner
07.06.2012	1.2	Persönlicher Bericht rwaltens	rwaltens
07.06.2012	1.3	Persönlicher Bericht dsteiner	dsteiner
08.06.2012	1.5	Persönlicher Bericht dschöttl	dschöttl
08.06.2012	1.6	Review	rwaltens



11.4 Allgemein

11.4.1 Zweck des Dokumentes

Dieses Dokument soll einen objektiven Rückblick über das Projekt bieten und sowohl die persönlichen Erfahrungen wie auch eine Aufwandanalyse beinhalten.

11.4.2 Gültigkeitsbereich

Dieses Dokument gilt als Grundlage des Projektes und ist daher über die gesamte Projektdauer gültig (20.02 bis 07.06.2012).

11.4.3 Definitionen und Abkürzungen

Siehe „Glossar“.



11.5 Methoden & Technologien

Für EistCockpit wurde aufgrund der speziellen, zeitlichen Situation ein Hybridvorgehensmodell aus RUP und Scrum verwendet. Das bedeutet, dass während den ersten 4 Wochen für die Aufnahme der Anforderungen und die Vorstudie mehrheitlich ein RUP ähnliches Vorgehen benutzt wurde, während in der Entwicklungszeit mehr nach Scrum gearbeitet wurde.

In den 4 Wochen der Anforderungsaufnahme konnten zwar alle gesteckten Ziele erreicht werden, das dadurch entstandene Loch im Zeitbudget hatte aber noch bis zum Ende des Projektes Auswirkungen. Der späte Startschuss für die Entwicklung der Prototypen sorgte für kaskadierende Verzögerungen und schlussendlich dafür, dass nicht alle Wunschfeatures umgesetzt werden konnten.

Die Scrum Methodik wurde leider nicht konsequent durchgezogen, so wurden beispielsweise im Sprint 2 die vorgegebene Laufzeit deutlich überschritten und der Sprint wurde zu spät abgeschlossen.

Als äussert hilfreich haben sich die regelmässigen Meetings mit dem Betreuer erwiesen. Nicht nur bekamen wir ein Feedback ob wir auf dem richtigen Weg sind, sondern waren es auch Kontrollpunkte für uns die eine gewisse Konsistenz ins Schaffen brachte.

Mit dem .net Framework und WPF bekamen wir solide und mächtige Tools in die Hand, mit denen sich unsere Vorstellungen der Applikation relativ leicht umsetzen liessen. Innerhalb des Projektteam zeigte sich ein breites Erfahrungsspektrum mit dieser Technologie, die von keiner, über wenig bis zu viel Praxiswissen reichte. Das machte die Abschätzung von Zeitkontingenten zum Teil schwierig.

Redmine erwies sich als sehr pragmatisch und gut geeignet für solche Projekte.



11.6 Persönliche Berichte

11.6.1 David Schöttl

Vor einem Jahr, im Juni 2011, während meines zweiten WKs als Tech Fw im Stab des Ristl Bat 20, ersann ich das System 'EistCockpit'. Und das ging so:

Mein Zellenchef S3 Hptm Klötzli befahl mir, ich solle mir die Sache mit dem 'IMFSViewer'[40] "anzuschauen". Gemeint hatte er wohl, ich solle die Software auf einem alten Armee Laptop installieren. Verstanden hatte ich aber, ich solle mir den IMFSViewer unter die Lupe nehmen, um daraus eine brauchbare Software zu machen...

Ich schaute mir das Innenleben von IMFSViewer an und kam schnell zum Schluss, dass eine neue Software entwickelt werden musste.

Zuerst konzentrierte ich meine Gedanken nur auf eine verbessert Netzübersichts-Software. Während der täglichen Arbeit sowie bei Gesprächen mit anderen Mitarbeitern jeder Stufe fiel aber bald auf, dass wir in der Eist Tm ein Software-System brauchen könnten, welches einen sehr viel breiteren Funktionsumfang aufweist, als IMFSViewer... Die Idee für 'EistCockpit' war geboren!

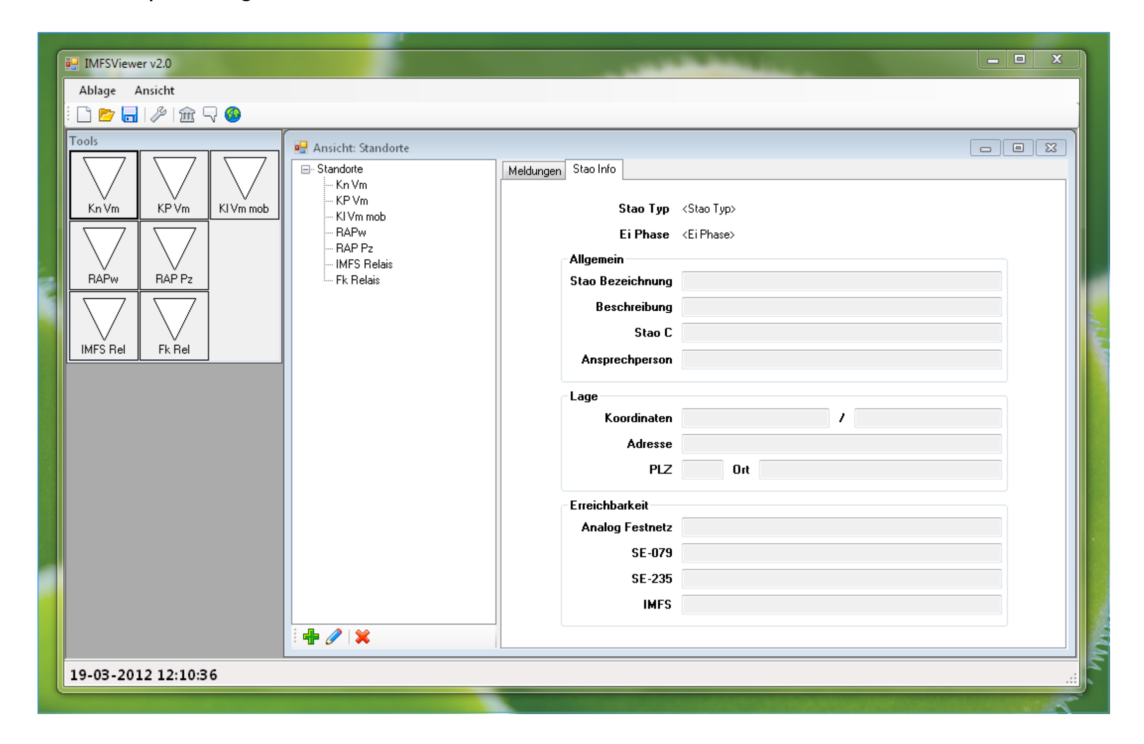


Abbildung 97 IMFSViewer 2.0 POC, entstanden im WK 2011; Ideen für EistCockpit sind am entstehen (Services Plugin, Messages Plugin, Locations Plugin).

Vom Umfang her war mir klar, dass dieses System niemals während eines WKs implementiert werden könnte – selbst wenn ich die ganze Zeit über dafür abdetachiert wäre. Als Thema für die anstehenden Semester- und Bachelorarbeiten jedoch schien das Projekt perfekt. Gegen Ende des WK 2011 hatte ich die Gelegenheit, mit dem Kdt Stv Maj

[40] Im WK 2010 bekam ich dieses Stück "Software" zum ersten Mal zu Gesicht, musste mich aber damals glücklicher Weise nicht damit abgeben: Dieses Unding, welches eine Netzübersicht darstellt, besteht aus einer Microsoft Access Datenbank; Benutzereingaben haben nicht in einem UI, sondern direkt auf den Datenbanktabellen zu erfolgen – inkl. der Koordinaten für die Symbole in der Netzübersicht... Gemäss ReadMe.txt File wurde dieser Haufen aus VisualBasic Code von einem Soldaten während des WK 2005 verbrochen.



Veraguth über die Projektidee zu sprechen, und er bot schliesslich an, als Auftraggeber für die SA 'EistCockpit' aufzutreten.

Wie es der Zufall wollte, lernte ich am ersten Tag des WK 2011 den damaligen Soldaten Diego Steiner persönlich kennen; ich kannte ihn bislang nur vom Sehen her von der HSR und wusste nicht, dass er auch im RistlBat 20 eingeteilt war.[41]

Als ich das SA Thema 'EistCockpit' schliesslich bei der HSR eingegeben habe, musste ich mich nach Projektpartnern umschauen. Diego Steiner schien mir – mitunter aufgrund seiner Einteilung im Ristl Bat 20 – ein passender Kandidat zu sein, also sprach ich ihn spontan darauf an, ob er bei dem Projekt dabei wäre. Er war sofort von der Idee überzeugt und brachte zusätzlich Remo Waltenspül mit ins Team.

Der WK 2012, welcher genau auf den Semesteranfang FS2012 – und damit auch auf den Beginn der Semesterarbeit – fiel, kam uns insofern gelegen, weil er die Möglichkeit bot, vier volle Wochen "beim Kunden" zu verbringen – um Requirements zu sammeln und Benutzerbefragungen durchzuführen. [42]

Als wir im Team die Arbeiten für den WK definierten, gingen wir davon aus, wir hätten dafür genügend Zeit zur Verfügung. Unsere Annahmen waren leider etwas zu optimistisch: Es war wohl voraussehbar, dass ich in meiner Funktion als Tech Fw in der Eist Tm nicht viel Zeit erübrigen kann – mit einer Umteilung von Sdt Steiner in die Kanzlei und dem forderndsten WK der letzen Jahre hatten wir jedoch nicht gerechnet.[43] Wir können uns daher sehr glücklich schätzen, dass unser Aufkl Sdt Waltenspül – einem Aufklärer fehlt das nötige Fachwissen im Bereich Richtstrahl – auf der Wache des Stabs KP eingeteilt wurde und so einen geregelten Tagesablauf hatte. Dadurch konnte er ausreichend Zeit für die Semesterarbeit aufbringen, um die "Ausfälle" von Diego und mir auszugleichen.[44]

Nach dem WK begann dann die wirkliche Arbeit am Projekt. SCRUM hatte ich bisher noch nie direkt in einem Projekt angewandt, weshalb ich mich zuerst etwas an diese Arbeitsweise gewöhnen musste. Ebenfalls neu für mich war die Projektumgebung 'Redmine', da ich für meine anderen Projekte 'TRAC' verwende.

Mir wurde bald klar, dass der vorgenommene Feature-Umfang nicht im Rahmen der SA vollständig umgesetzt werden kann. Dies war aber auch so angedacht: Im Rahmen der SA werden die nötigen Vorarbeiten wie Benutzerbefragung, Requirements Definition, etc. behandelt und ein erster Teil des Systems implementiert. Im Rahmen der BA wird das System verbessert, erweitert und zu einem "kriegstauglichen" System komplettiert.

Nachdem die Prototypen erstellt waren und wir wussten, wie wir bei der Implementation vorgehen wollen, ging die Entwicklung relativ zügig voran. In den seltenen Fällen, in denen Probleme auftauchten, konnten diese jeweils innert kurzer Zeit gelöst werden.

Die vier "fehlenden" Wochen – verursacht durch den WK – waren während der ganzen Projektarbeit in Form von notorischem Zeitmangel dauernd spürbar und haben sichtbare Spuren in der aktuellen Version von EistCockpit hinterlassen: Oft wurden Entscheide für Projektstrukturen, Namensgebung und dergleichen zu schnell oder einfach aus Zeitdruck

---

[41] Cool, damit waren wir schon drei Informatiker von der HSR im Ristl Bat 20 bzw. in dessen Stabs Kp: Gfr Thomas Corbat aus der Stabskanzlei, Ik Pi Sdt Diego Steiner und Tech Fw David Schöttl!
[42] Für Sdt Waltenspül wurde sogar extra innert drei Tagen ein Gast-WK bei uns im Ristl Bat 20 organisiert, damit er dabei sein konnte.
[43] Sdt Steiner hat seine Arbeit, den Umständen zum Trotz, sehr gut gemacht: Er wurde zum Obgfr befördert und wird wohl noch lange der Kanzlei erhalten bleiben... ;)
[44] Remo: We salute you!



gefällt. Auch wenn diese später als suboptimal erkennbar wurden, so war keine Zeit für ein entsprechendes Refactoring verfügbar – schliesslich ist man schon mit dem aktuellen Feature in Verzug...

Die wöchentlichen Team Collaboration Meetings waren sehr wertvoll, weil sich so das Team bei den Entscheidungen beraten, gegenseitig Hinweise/Feedback geben, und das weitere Vorgehen planen konnte. Ebenfalls wertvolle Inputs für das Projekt wurden während den Meetings mit dem Betreuer der Arbeit, Thomas Corbat, generiert. Für die BA sollte jedoch mehr Zeit für die Team Collaboration eingerechnet werden, da sich der gegenseitige Ideenaustausch und Absprache positiv auf die Qualität der Code Base auswirken.

Ich bin zufrieden und erfreut: Heute, ein Jahr, nachdem die Idee für EistCockpit entstand, haben wir eine erste Version des System erschaffen – eine solide Grundlage bzw. ein Framework, welches nun im Rahmen der BA verfeinert/erweitert wird und womit weitere Features für EistCockpit implementiert werden können.

Mein besonderer Dank gilt meinen beiden Teampartnern Diego Steiner und Remo Waltenspül, ohne deren grossen Einsatz das Projekt nicht hätte umgesetzt werden können.

Ich freue mich darauf, EistCockpit im Rahmen der BA – und in der darauf folgenden Zeit – weiterentwickeln zu können.



11.6.2 Remo Waltenspül

Etwa drei Monate vor dem ersten Kick-Off Meeting weihte uns David Schöttl in sein Vorhaben ein, die Studienarbeit im Auftrag der Schweizer Armee durchzuführen. Anfänglich war ich noch nicht so überzeugt, da die vorgeschlagene Programmiersprache .NET C# mir noch nicht wirklich geläufig war. Doch nach reichlicher Überlegung bin ich zum Schluss gekommen, dass dies zwar eine Herausforderung darstellt, jedoch aufgrund des während dem letzten Semester angeeigneten Wissens aus dem Modul Microsoft Technologien realistisch sein sollte. Das zum Teil fehlende Knowhow bzw. der routinierte Umgang versuchte ich mit einem grösseren Arbeitseinsatz wett zu machen. Deshalb resultierte ein Arbeitsaufwand der deutlich über dem erwarteten Wert liegt.

Während der in das Projekt investierten Zeit, habe ich viele Erfahrungen sammeln können, speziell im Umgang mit dem .NET Framework 4.0 sowie im Bereich des Projektmanagements mit SCRUM. Zudem konnte ich bei kniffligen Problemen oder Bugs, die ich selber auch nach längerem recherchieren und analysieren nicht lösen konnte, auf ein versiertes, erfahrenes Team zurückgreifen. Das war für mich sehr wertvoll, da ich durch die Inputs von meinen beiden Teamkameraden immer wieder etwas lernen konnte, auch wenn es nur neue Ansätze waren ein Problem zu lösen.

Gut, fand ich auch die fest eingeplanten Teammeetings am Montagnachmittag, da man dort das gemeinsame weitere Vorgehen besprechen sowie allfällige Schwierigkeiten zusammen diskutieren konnte. Auf der anderen Seite hätte ich es geschätzt, wenn es mehr gemeinsame Arbeitszeiten gegeben hätte. Das ist womöglich auch eine Ursache für einige kleinere Inkonsistenzen, da mehrheitlich die Arbeit unabhängig voneinander verrichtet wurde. Dies führte in meinem Fall auch zu grösseren Refactorings, da zum Beispiel erst relativ spät erkannt wurde, dass eine Klasse zu viele Funktionen übernahm und daraus eine tiefe Kohäsion resultierte.

Ein spannender Teil der Studienarbeit war sicherlich der vierwöchige Wiederholungskurs der Schweizer Armee in Beckenried, welcher mir einen Einblick in die Abläufe der Triage, Eist Tm ermöglichte. Während dieser Dienstleistung konnten wir direkt vor Ort die Anforderungen an das zu entwickelnde System mithilfe von Befragungen aufnehmen. Diese Arbeit hat mir aufgrund der Interaktion mit potentiellen Benutzern sehr zugesagt und war eine gute Motivation für den Einstieg ins Projekt.

Für eine weitere Arbeit würde ich schon von Beginn an schauen, dass die Ziele gemeinsam definiert werden und das man sich untereinander besser abspricht. Weiter müsste man in einer Folgearbeit sich konsequenter an die Grundsätze von SCRUM halten, so sollten zum Beispiel Sprints nicht künstlich verlängert werden.

Abschliessend würde ich das Projekt trotzdem als Erfolg werten, auch wenn einzelne Teile der Studienarbeit nicht zu meiner vollsten Zufriedenheit realisiert wurden.



11.6.3 Diego Steiner

Da ich mich noch nicht für ein Thema für die Studienarbeit entschieden hatte, kam mir die Einladung das EistCockpit mit David Schöttl und Remo Waltenspül zu verwirklichen gerade recht.

Die Möglichkeit, die SA mit dem WK zu verbinden fand ich ideal, aus eigener Erfahrung boten solche Wiederholungskurse mehr als genug Zeit um an einem eigenen Projekt zu arbeiten. Als es dann aber losging wurde mir langsam bewusst, dass es diesmal wohl anders sein würde. Ich wurde nicht wie üblich bei den Informatikpionieren eingeteilt, sondern in der Stabskanzlei. Die Arbeiten in der Kanzlei nahmen fast während des ganzen WKs meine vollständige Aufmerksamkeit in Anspruch, schliesslich hiess es zuerst den primären Auftrag zu erfüllen, bevor am EistCockpit gearbeitet werden konnte. Mit viel Energie und dem starken Einsatz von Remo gelang es uns aber dennoch, die für den WK gesteckten Ziele zu erreichen.

Wieder an der HSR, galt es die gewonnenen Erkenntnisse in ein Produkt umzusetzen. Das gleichzeitige Besuchen des User Interfaces 2 Modul stellte sich als eine riesige Hilfe heraus. Die Grundvorstellungen waren klar, so konnten wir rasch mit der Entwicklung der Prototypen fortfahren. Trotzdem dass meine .net 2.0 Kenntnisse ein bisschen eingerostet waren, kam ich schnell auf ein gutes Ergebnis mit dem SQLite-Prototyp.

Als es dann im Sprint 2 an die Entwicklung des echten Produktes ging, schien am Anfang Alles in den vorgesehenen Bahnen zu verlaufen; Wir kamen gut voran, doch gegen Ende des Sprints wurde langsam klar, dass wir den definierten Zeitrahmen nicht einhalten werden können. Spätestens an diesem Punkt hätte ich als Scrum-Master für das Projekt eingreifen müssen. Hätten wir eine Iteration mehr gehabt, wäre es möglich gewesen diese Umstände für den nachfolgenden Sprint zu beachten und die Anzahl der User Stories zu minimieren. Doch ich war ja selbst auch stark mit der Entwicklung beschäftigt und damit ist mir inzwischen klar geworden, wieso der Scrum-Master nicht selbst Teil des Entwicklerteams sein sollte. So verzögerte sich das Ende des Sprints und damit auch der Anfang des Nächsten.

Obwohl wir mit viel Einsatz unsere Ziele doch noch erreichen konnten, würde ich das nächste Mal versuchen das definierte Vorgehensmodell ohne Kompromisse durchzusetzen und verstärkt auf ein klar strukturiertes Projektmanagement achten. So liessen sich einige Probleme in der Zukunft vermeiden.

Alles in Allem hat mir die Studienarbeit in diesem Team Freude bereitet und ich konnte viele Dinge, sowohl in technischer als auch in Hinsicht auf Vorgehen, für mich persönlich mitnehmen.



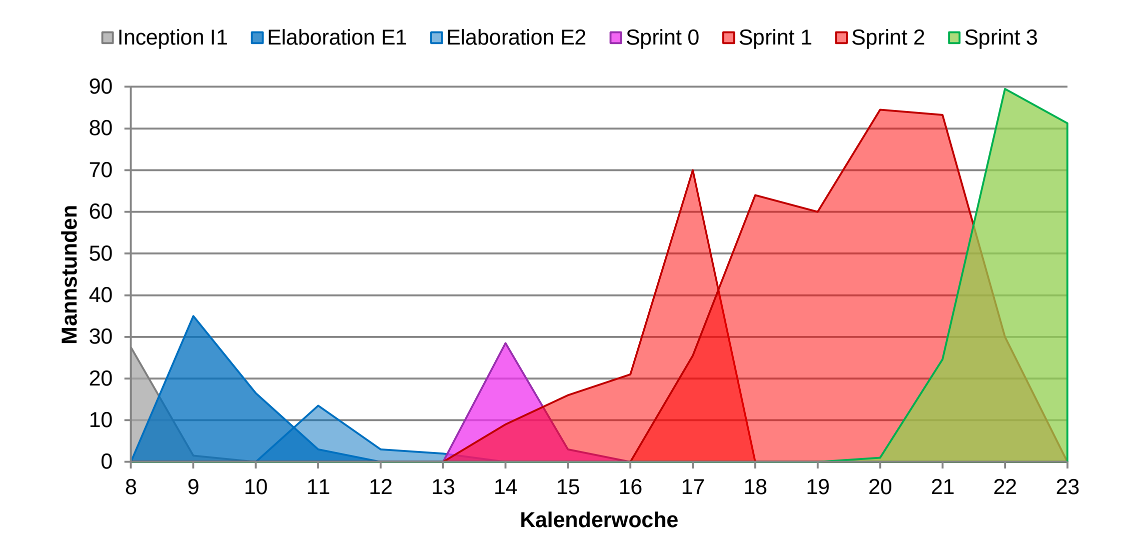
11.7 Aufwandanalyse

11.7.1 Phasen & Sprints

In diesem Abschnitt sind die Aufwände nach Sprints aufgesplittert. Im untenstehenden Diagramm ist die Verteilung der Aufwände über die Wochen ersichtlich.

Phase/Sprint	Geschätzter Aufwand	Tatsächlicher Aufwand	Anzahl Tickets
Inception I1	31h	29h	7h
Elaboration E1	55h	54.5h	14h
Elaboration E2	30h	18.5h	5h
Sprint 0	34h	32h	7h
Sprint 1	126h	116h	18h
Sprint 2	319h	347.25h	45h
Sprint 3	219.5h	180h	34h
Total	**814.5h**	**777.25h**	**130h**

Tabelle 70 Aufwand pro Phase/Sprint



11.7.1.1 Phasen

Der Übersicht halber wurden hier die ersten 3 Phasen zu einem Block zusammengefasst. Dieser Block umfasst die Phase „Inception I1", die Phase „Elaboration E1" sowie die Phase „Elaboration E2"

Dauer	5 Wochen
Geschätzter Aufwand	116h
Tatsächlicher Aufwand	102h
Tickets	26



Tickets

#	Aktivität	Status	Prio.	Name	Verantwortlich	Phase
42	PM-QM	CLO	Normal	Reservezeit Risiken	Remo Waltenspül	Inception I1
34	Collaboration	CLO	Normal	Sitzung 24.02.2012	Alle	Inception I1
33	PM-QM	CLO	Normal	Infrastuktur einrichten	Diego Steiner	Inception I1
28	Collaboration	CLO	Normal	Sitzung 22.02.2012	Alle	Inception I1
27	Collaboration	CLO	Normal	Sitzung 20.02.2012	Alle	Inception I1
17	Documentation	Resolved	Normal	Wiki Work	David Schöttl	Inception I1
1	PM-QM	CLO	Normal	Reviews I1	Alle	Inception I1
43	PM-QM	REJ	Normal	Reservezeit Risiken	Remo Waltenspül	Elaboration E1
41	Documentation	CLO	Normal	Risikomanagement erstellt	Remo Waltenspül	Elaboration E1
40	Documentation	CLO	Normal	Glossar erstellen	Remo Waltenspül	Elaboration E1
39	Documentation	CLO	Normal	Interview auswerten	Remo Waltenspül	Elaboration E1
32	Collaboration	REJ	Normal	Sitzung 12.03.2012	Alle	Elaboration E1
31	Collaboration	CLO	Normal	Sitzung 08.03.2012	Alle	Elaboration E1
25	PM-QM	CLO	Normal	Sitzungstickets erfassen und bisherige Stunden buchen	Diego Steiner	Elaboration E1
24	Documentation	CLO	Normal	Personas verfassen	Diego Steiner	Elaboration E1
23	Documentation	CLO	Normal	Interviews durchführen	Diego Steiner & Remo Waltenspül	Elaboration E1
22	Documentation	CLO	Normal	Interviewsfragen verfassen	Diego Steiner & Remo Waltenspül	Elaboration E1
16	Collaboration	CLO	Normal	Sitzung 05.03.2012 19:00	Alle	Elaboration E1
15	PM-QM	CLO	Normal	Matrix of Doom (MoD) - Featureabdeckung	Diego Steiner	Elaboration E1
14	Collaboration	CLO	Normal	Sitzung 05.03.2012 14:00	Alle	Elaboration E1
2	PM-QM	CLO	Normal	Reviews E1	Alle	Elaboration E1
48	PM-QM	REJ	Normal	Reservezeit Risiken	Remo Waltenspül	Elaboration E2
46	Documentation	CLO	Normal	Beobachtung Triage	Remo Waltenspül	Elaboration E2
45	Documentation	CLO	Normal	Visionsdokument erstellen	Remo Waltenspül	Elaboration E2
44	Documentation	REJ	Normal	Storyboard	Diego Steiner	Elaboration E2
3	PM-QM	CLO	Normal	Reviews E2	Alle	Elaboration E2

Tabelle 71 Tickets der ersten drei Phasen

Auswertung

Nachdem die Vorbereitungswoche relativ speditiv ablief, waren die 4 Wochen des Wiederholungskurses nicht besonders produktiv. Allerdings musste mit diesem Risiko gerechnet werden, weil dem Projektteam keine freie Arbeitszeit garantiert werden konnte. Trotz dieses Rückschlages konnten in diesen Phasen alle wichtigen Ziele erreicht werden.



11.7.1.2 Sprint 0

Dauer	1 Woche
Geschätzter Aufwand	34h
Tatsächlicher Aufwand	32h
Tickets	7

Tickets

#	Aktivität	Status	Prio.	Name	Verantwortlich	Phase
50	Collaboration	CLO	Normal	Sitzung 28.03.2012	Alle	Sprint 0
49	Collaboration	CLO	Normal	Sitzung 26.03.2012	Alle	Sprint 0
36	PM-QM	CLO	Normal	Backlog priorisieren	David Schöttl	Sprint 0
35	PM-QM	CLO	Normal	Project Backlog erstellen	David Schöttl	Sprint 0
18	PM-QM	CLO	Normal	Reviews SP0	Alle	Sprint 0
52	PM-QM	CLO	Normal	Sitzung 28.03.2012 vorbereiten	Diego Steiner	Sprint 0
51	Documentation	CLO	Normal	Plugin Ideas	David Schöttl	Sprint 0

Tabelle 72 Tickets zu Sprint 0

Auswertung

Nach dem schwachen Start konnte das Projektteam sich nun die verbleibenden Wochen mit den User Stories einteilen, die sich in der Vorstudie angesammelt hatten.



11.7.1.3 Sprint 1

Dauer	2 Wochen
Geschätzter Aufwand	126h
Tatsächlicher Aufwand	116h
Tickets	18

Tickets

#	Aktivität	Status	Prio.	Name	Verantwortlich	Phase
75	Feature	CLO	Normal	Style von Buttons laden	Remo Waltenspül	Sprint 1
74	Feature	CLO	Normal	Dynamisch während Laufzeit Plugins laden	Remo Waltenspül	Sprint 1
73	Feature	CLO	Normal	Test UI entwerfen	Remo Waltenspül	Sprint 1
72	Support	CLO	Normal	Einarbeiten in WPF	Remo Waltenspül	Sprint 1
71	Feature	CLO	Normal	UI Plugin Prototyp erstellen	Remo Waltenspül	Sprint 1
70	Feature	CLO	Normal	Models mittels Entity-Framework umsetzen	Diego Steiner	Sprint 1
69	Documentation	CLO	Normal	Dokumente überarbeiten	Remo Waltenspül	Sprint 1
68	Feature	CLO	Normal	DLL Plugin Prototype	Remo Waltenspül	Sprint 1
67	Feature	CLO	Normal	SQLite Adapter implementieren	Diego Steiner	Sprint 1
66	Feature	CLO	Normal	WCF Prototyp bauen	David Schöttl	Sprint 1
65	Documentation	CLO	Normal	Schichtenarchitektur überarbeiten	Remo Waltenspül	Sprint 1
64	Feature	CLO	Normal	Domain Model erstellen	Diego Steiner	Sprint 1
63	Documentation	CLO	Normal	Schichtenarchitektur dokumentieren	Remo Waltenspül	Sprint 1
62	Collaboration	CLO	Normal	Sitzung 02.04.2012	Remo Waltenspül	Sprint 1
61	Support	CLO	Normal	Projekt aufsetzen	Remo Waltenspül	Sprint 1
53	Feature	CLO	Normal	Prototypen EistCockpit (PoC)	Alle	Sprint 1
38	Documentation	CLO	Normal	Paper Prototype erstellen und durchspielen	Alle	Sprint 1
19	PM-QM	CLO	Normal	Reviews SP1	Alle	Sprint 1

Tabelle 73 Tickets zu Sprint 1

Auswertung

In diesem Sprint standen die Entwicklung des User Interfaces und der technischen Prototypen sowie die Einarbeitung in die Technologie im Vordergrund. Weil die technische Vorgehensweise noch ein wenig unklar war, gab es am zu Beginn Schwierigkeiten.



11.7.1.4 Sprint 2

Dauer	5 Wochen
Geschätzter Aufwand	319h
Tatsächlicher Aufwand	347.25h
Tickets	45

Tickets

#	Aktivität	Status	Prio.	Name	Verantwortlich	Phase
150	Feature	CLO	Normal	Usability Funktionen implementieren	Remo Waltenspül	Sprint 2
149	Collaboration	CLO	Normal	Sitzung 23.05.2012	Remo Waltenspül	Sprint 2
124	Feature	CLO	Normal	UI überarbeiten	Remo Waltenspül	Sprint 2
123	Feature	CLO	Normal	Validierung	Remo Waltenspül	Sprint 2
122	Collaboration	CLO	Normal	Team Sitzung 14.05.12	Remo Waltenspül	Sprint 2
121	Collaboration	CLO	Normal	Sitzung 09.05.2012	Remo Waltenspül	Sprint 2
120	Feature	CLO	Normal	Plugin Grundgerüst erstellen	Diego Steiner	Sprint 2
118	Feature	CLO	Normal	Event Handling, Speichern	Remo Waltenspül	Sprint 2
117	Feature	CLO	Normal	Data Binding	Remo Waltenspül	Sprint 2
116	Feature	CLO	Normal	Location UI erstellen	Remo Waltenspül	Sprint 2
115	Support	CLO	Normal	TestClient	David Schöttl	Sprint 2
114	Support	CLO	Normal	ExampleDataGenerator	David Schöttl	Sprint 2
113	Bug	CLO	Immediate	Generated DTOs Issue	David Schöttl	Sprint 2
112	Documentation	CLO	Normal	Doxygen Documentation Guide	David Schöttl	Sprint 2
111	PM-QM	CLO	Normal	Projektmanagement SP2	Alle	Sprint 2
108	Collaboration	CLO	Normal	Team Sitzung 07.05.12	Remo Waltenspül	Sprint 2
107	Feature	CLO	High	Message Dispatching	Diego Steiner	Sprint 2
106	Feature	CLO	Low	Message Drafts	Diego Steiner	Sprint 2
105	Feature	CLO	High	Filter	Diego Steiner	Sprint 2
104	Feature	CLO	Normal	Message Index/Read	Diego Steiner	Sprint 2
103	Feature	CLO	High	UI Create Message	Diego Steiner	Sprint 2
102	Documentation	CLO	Normal	Review Architecture	David Schöttl	Sprint 2
101	Collaboration	CLO	Normal	Team Sitzung 30.04.12	Remo Waltenspül	Sprint 2
100	Feature	CLO	Normal	Plugin-Mapping	Remo Waltenspül	Sprint 2

Tabelle 74 Tickets zu Sprint 2

99	Documentation	CLO	Normal	Kognitiver Test dokumentiert	Remo Waltenspül	Sprint 2
98	Collaboration	CLO	Normal	Team Sitzung 23.04.12	Remo Waltenspül	Sprint 2
97	PM-QM	CLO	Normal	Code Refactoring	Remo Waltenspül	Sprint 2
96	Documentation	CLO	Normal	Papier-Prototyp auswerten	Remo Waltenspül	Sprint 2
94	Collaboration	CLO	Normal	Sitzung 25.04.2012	Remo Waltenspül	Sprint 2
93	Feature	CLO	Normal	UI Operation Configuration	David Schöttl	Sprint 2
92	Feature	CLO	Normal	UI Service Overview	Diego Steiner	Sprint 2
91	Feature	CLO	Normal	XML Konfigurationsdatei	David Schöttl	Sprint 2
88	Feature	CLO	Normal	Tutorial WPF	Remo Waltenspül	Sprint 2
87	Feature	CLO	Normal	Grundgerüst erstellen	Remo Waltenspül	Sprint 2
86	Feature	CLO	Normal	DTO erstellen	David Schöttl	Sprint 2
85	Feature	CLO	Normal	DAL Zugriffsmethoden	David Schöttl	Sprint 2
84	Feature	CLO	Normal	Webservice erstellen	David Schöttl	Sprint 2
82	Feature	CLO	High	Message Flow	Diego Steiner	Sprint 2
81	Feature	CLO	Normal	Locations	Remo Waltenspül	Sprint 2
80	Feature	CLO	Normal	Operation Configuration	David Schöttl	Sprint 2
79	Feature	CLO	Normal	Service Overview	Diego Steiner	Sprint 2
78	Feature	CLO	Normal	Application Configuration	David Schöttl	Sprint 2
77	Feature	CLO	Normal	EistCockpit	Remo Waltenspül	Sprint 2
76	Feature	CLO	Normal	EistCockpitDAL	David Schöttl	Sprint 2
20	PM-QM	CLO	Normal	Reviews SP2	Alle	Sprint 2

Auswertung

Dieser Sprint stand unter dem Stern „Das aus den Prototypen gelernte umsetzen". Nachdem nun alle Projektmitglieder eingearbeitet waren, ging es rasch vorwärts. Leider unterschätzten wir einige Aufgaben, was dazu führte dass sich der Sprint nach hinten in die Länge zog.

Ziel des Sprints war es, eine lauffähige Betaversion der Applikation fertigzustellen, was dann mit ein wenig Verspätung auch gelang.



11.7.1.5 Sprint 3

Dauer	3 Wochen
Geschätzter Aufwand	219h
Tatsächlicher Aufwand	180h
Tickets	34

Tickets

#	Aktivität	Status	Prio.	Name	Verantwortlich	Phase
164	Documentation	Resolved	Normal	Poster	David Schöttl	Sprint 3
163	PM-QM	Resolved	Normal	Bugs beheben	Remo Waltenspül	Sprint 3
162	PM-QM	Resolved	Normal	Refactoring	Remo Waltenspül	Sprint 3
161	PM-QM	Resolved	Normal	Code Dokumentation	Remo Waltenspül	Sprint 3
160	PM-QM	Resolved	Normal	Tests schreiben	Remo Waltenspül	Sprint 3
159	Feature	Resolved	Normal	Locations	Remo Waltenspül	Sprint 3
157	Collaboration	Resolved	Normal	Sitzung 06.06.2012	Remo Waltenspül	Sprint 3
156	Documentation	Resolved	Normal	Code Dokumentation	Alle	Sprint 3
155	Feature	Resolved	Normal	Unit Tests	Alle	Sprint 3
154	Documentation	Resolved	Normal	Vision	Diego Steiner	Sprint 3
153	PM-QM	In Progress	Normal	Projektmanagement SP3	Alle	Sprint 3
152	Documentation	Resolved	Normal	Dokumentenbasis erstellen	Remo Waltenspül	Sprint 3
148	Documentation	Resolved	Normal	Entwurf	David Schöttl	Sprint 3
147	Documentation	New	Normal	Anhang	Remo Waltenspül	Sprint 3
146	Documentation	New	Normal	Projekt Retroperspektive	Diego Steiner	Sprint 3
145	Documentation	New	Normal	Schlussfolgerung	Remo Waltenspül	Sprint 3
144	Documentation	New	Normal	Prototypen	Remo Waltenspül	Sprint 3
143	Documentation	Resolved	Normal	Domain Analyse	Diego Steiner	Sprint 3
141	Documentation	Resolved	Normal	Management Summary	Remo Waltenspül	Sprint 3
139	Documentation	In Progress	Normal	Anforderungen Dokument	Diego Steiner	Sprint 3
138	Documentation	New	Normal	Entwicklerdokumentatio n	David Schöttl	Sprint 3
137	Documentation	Resolved	Normal	Test Dokument	Diego Steiner	Sprint 3

Tabelle 75 Tickets zu Sprint 3



136	Documentation	Resolved	Normal	Projektmanagement Dokument	Diego Steiner	Sprint 3
134	Documentation	Resolved	Normal	Extended Management Summary	Remo Waltenspül	Sprint 3
133	Documentation	Resolved	Normal	Abstract erstellen	Remo Waltenspül	Sprint 3
132	Documentation	Resolved	Normal	Vorstudien erweitern	Remo Waltenspül	Sprint 3
131	Documentation	New	Normal	Dokumentation	Alle	Sprint 3
130	Feature	Resolved	Normal	Message Model im Plugin	Diego Steiner	Sprint 3
129	Feature	REJ	Normal	Polling Gateway einrichten	Diego Steiner	Sprint 3
128	Feature	Resolved	Normal	Refactoring	Alle	Sprint 3
127	Feature	Resolved	High	Client Synchronization	Diego Steiner	Sprint 3
125	Feature	REJ	Normal	Validation	Remo Waltenspül	Sprint 3
90	Feature	Resolved	Normal	UI Role Mapping	David Schöttl	Sprint 3
47	Documentation	Resolved	Normal	Logo erstellen	David Schöttl	Sprint 3
21	PM-QM	New	Normal	Reviews SP3	Alle	Sprint 3

Auswertung

Im letzten Sprint galt es noch die Bugs auszumerzen, die sich noch im Prototyp befanden und dann zur Fertigstellung der Dokumentation zu schreiten. Der Sprint zeichnete sich vor allem dadurch aus, dass Alle Projektmitglieder nochmals bedingungslosen Einsatz zeigten.

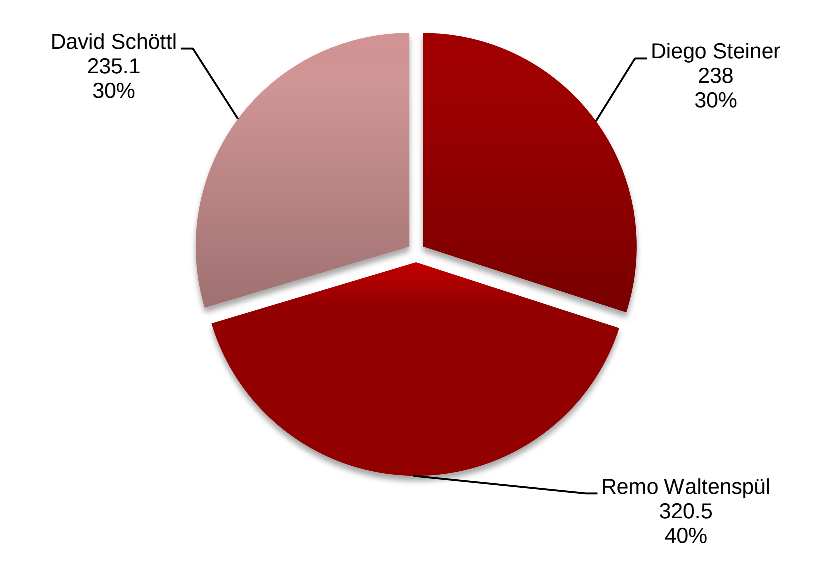


11.7.2 Personen

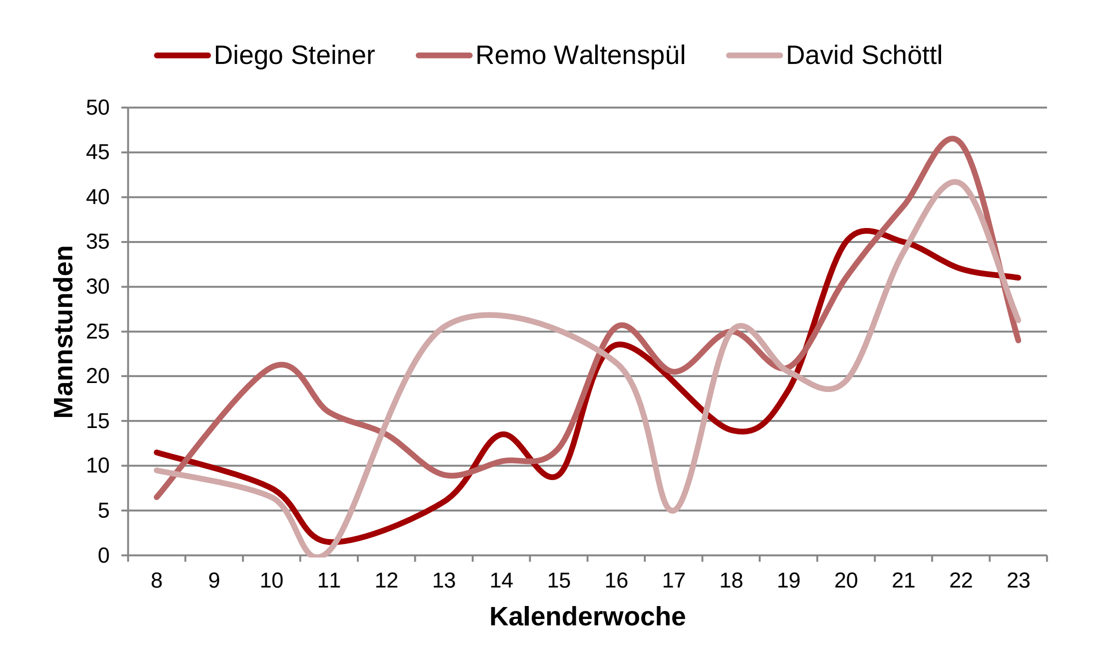
In der nachfolgenden Tabelle und Diagrammen ist der Arbeitsaufwand pro Person beschrieben.

Diego Steiner	238.00	30%
Remo Waltenspül	320.50	40%
David Schöttl	235.10	30%
Total	**793.60**	**100%**

Tabelle 76 Auflistung Arbeitszeiten



Auffällig hierbei ist sicher die nicht ganz gleichmässige Aufwandsverteilung zwischen Remo Waltenspül und den anderen Projektmitgliedern. Dies lässt sich zum Teil mit dem Erfahrungsunterschied erklären.



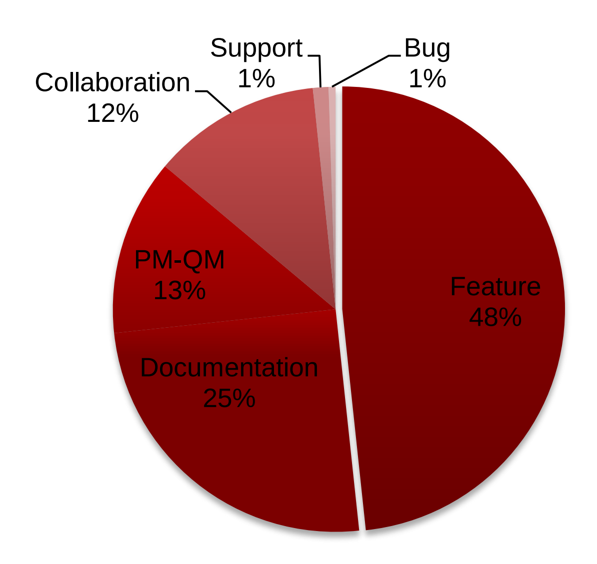


11.7.3 Tätigkeiten

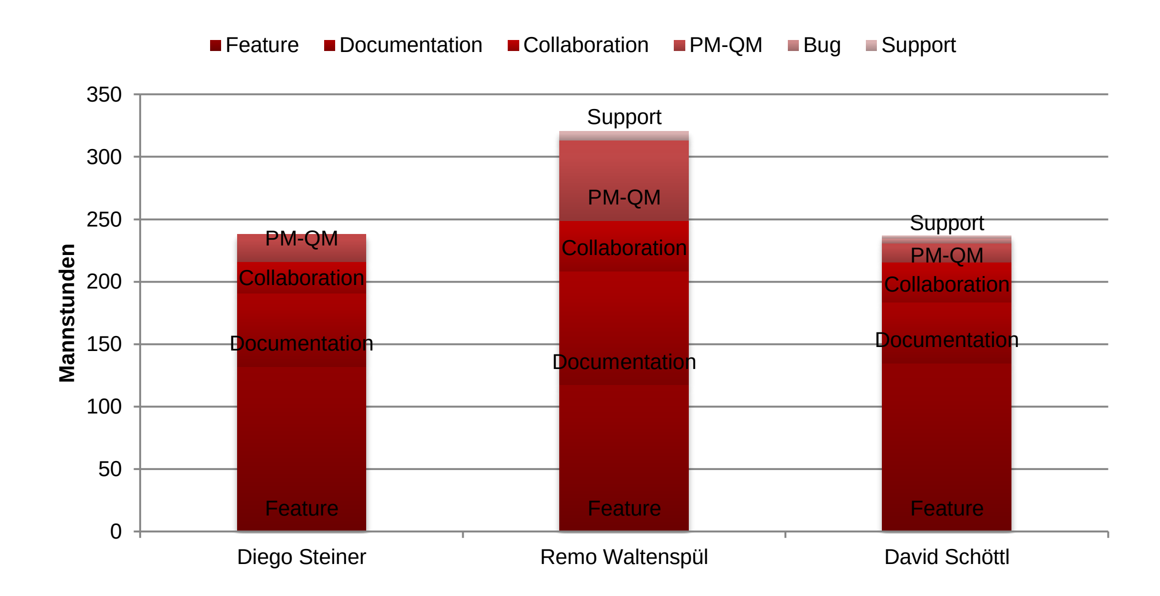
In diesem Abschnitt werden die Aufwände anhand der Tätigkeiten untersucht.

Feature	384.35h	48%
Documentation	198.75h	25%
Projekt- und Qualitätsmanagement	102h	13%
Collaboration	97.5h	12%
Support	9h	1%
Bug	4h	1%
Total	**795.6h**	**100%**

Tabelle 77 Auflistung Arbeitszeiten pro Tätigkeit



Rund die Hälfte der Projektzeit wurde in die Entwicklung selbst gesteckt, während ein Viertel für die Dokumentation aufgebracht werden musste.





EistCockpit

VI. Anhang

David Schöttl, Remo Waltenspül & Diego Steiner

EistCockpit

Anhang A

Verzeichnisse Referenzen

David Schöttl, Remo Waltenspül & Diego Steiner

A.1 Glossar

A.1.1 Begriffserklärung

Begriff	Beschreibung
Triage	Die Triage ist die primäre Anlaufstelle für Meldungen per Telefon, Fax etc.
Zelle	Eine Abteilung in der Zivilschutzanlage
Scrum	Vorgehensmodell zur Entwicklung von Software
Redmine	Projektmanagementsoftware
RUP	Rational Unified Process; Vorgehensmodell zur Entwicklung von Software
RESTful	Bezeichnet ein Programmierparadigma für Webanwendungen, URL stellt genau einen Seiteninhalt als Ergebnis einer serverseitigen Aktion dar.[1]
ADO .NET	Framework zur objektrelationen Abbildung
Product Backlog	Scrum Artefakt enthält alle User Stories für ein Projekt
Design Constraints	Bedingungen, Vorschriften für die Implementierung eines Systems
Requirements	Anforderungen
Redmine	Webbasiertes Projektmanagement Tool
Root Cause Analysis	Ursachen Analyse für bestimmte Probleme
Usability-Test	Benutzbarkeitstest
DRY	Dont Repeat yourself (Software Engineering Grundprinzip)
iOS	Apple Betriebssystem
Sqlite	Relationales Datenbanksystem
Code Conventions	Codier Standards

[1] [url16]: Representational state transfer, http://de.wikipedia.org/wiki/RESTful (04.06.2012)

A.1.2 Abkürzungserläuterung

Abkürzung	Beschreibung
AdA	Angehörige der Armee
ZSA	Zivilschutzanlage
KVK	Kadervorbereitungskurs
MoD	Matrix of Doom => Bewertungsraster für die Studienarbeit
Eist TM	Einsatzstelle Telematik
Ristl Bat 20	Richtstrahl Batallion 20
AD	Active Directory
XAML	Extensible Application Markup Language, ist eine Beschreibungssprache für die Oberflächengestaltung von Anwendungen
Stao	Standort
MEF	Microsoft Extensibility Framework
IMFS	Integriertes Militärisches Fernmeldesystem
Ristl Vrb	Richtstrahlverbindung
Ristl Kn	Richtstrahl Knoten
MVVM	Model View ViewModel
WPF	Windows Presentation Foundation
WCF	Windows Communication Foundation
MS	Microsoft
FGG	Führungsgrundgebiet
ApM	Anschläge pro Minute
TBZ	Telematik Bereitschaftszeiten
JSON	Javascript Object Notation
WSDL	Web Services Description Language
SOAP	Simple Object Access Protocol
CRUD	Create, Read, Update, Delete
DAL	Data Access Layer
DTO	Data Transfer Object
PoC	Proof of Concept

A.1.3 Quellenverzeichnis

[gloger11] Boris Gloger, "Scrum – Produkte zuverlässig und schnell entwickeln", 1. Auflage, Hanser Fachbuchverlag, ISBN 978-3-446-42524-8, 2011

[grechanik07] Mark Grechanik, Kevin M. Conroy, "Composing Integrated Systems Using GUI-Based Applications And Web Services", IEEE, International Conference on Services Computing (SCC 2007), 0-7695-2925-9/07, 2007

[heini12] Heini R., "Funktionale, nicht funktionale Anforderungen", http://www.anforderungsmanagement.ch/in_depth_vertiefung/funktionale_nicht_funktionale_anforderungen/, letzter Zugriff: 08.06.2012

[hiane09] Luiz Alexandre Hiane da Silva Maciel, Celso Massaki Hirata, "An Optimistic Technique for Transactions Control using REST Architectural Style", ACM, SAC'09 March 8-12, 978-1-60558-166-8/09/03, 2009

[hunt07] Lance Hunt, "C# Coding Standards for .NET", Document Version 1.15, Lance Hunt, http://www.lance-hunt.net, 2007, verfügbar unter: http://se.inf.ethz.ch/old/teaching/ss2007/251-0290-00/project/CSharpCodingStandards.pdf, letzter Zugriff: 08.06.2012

[url1] Wikipedia, "Führungsunterstützungsbrigade 41", Januar 2012, http://de.wikipedia.org/wiki/Führungsunterstützungsbrigade_41, letzter Zugriff: 08.06.2012

[url2] Wikipedia, "Don't repeat yourself", Mai 2012, http://en.wikipedia.org/wiki/DRY, letzter Zugriff: 08.06.2012

[url3] Wikipedia, "Dienstgrade der Schweizer Armee", Mai 2012, http://de.wikipedia.org/w/index.php?title=Dienstgrade_der_Schweizer_Armee&oldid=103606123, letzter Zugriff: 08.06.2012

[url4] Microsoft Corporation, "Windows User Experience Interaction Guidelines", September 2010, http://msdn.microsoft.com/de-ch/library/windows/desktop/aa511258.aspx, letzter Zugriff: 08.06.2012

[url5] Wikipedia, "Active Directory", Juni 2012, http://en.wikipedia.org/wiki/Active_Directory, letzter Zugriff: 28.05.2012

[url6] Wikipedia, "Java (programming language)", Juni 2012, http://en.wikipedia.org/wiki/Java_(programming_language), letzter Zugriff: 29.05.2012

[url7] Wikipedia, "C Sharp (programming language)", Juni 2012, http://en.wikipedia.org/wiki/C_Sharp_(programming_language), letzter Zugriff: 29.05.2012

[url8] Wikipedia, ".NET Framework", Juni 2012, http://en.wikipedia.org/wiki/.NET_Framework, letzter Zugriff: 03.06.2012

[url9] Microsoft Corporation, "Download: Microsoft .NET Framework 4.0 (eigenständiger Installer)", Februar 2011, http://www.microsoft.com/de-de/download/details.aspx?id=17718, letzter Zugriff: 03.06.2012

[url10] Wikipedia, "Windows Communication Foundation", Juni 2012, http://en.wikipedia.org/wiki/Windows_Communication_Foundation, letzter Zugriff: 04.06.2012

[url11] Microsoft Corporation, "What Is Windows Communication Foundation", 2012, http://msdn.microsoft.com/en-us/library/ms731082.aspx, letzter Zugriff: 04.06.2012

[url12] Wikipedia, "Windows Presentation Foundation", Juni 2012, http://en.wikipedia.org/wiki/Windows_Presentation_Foundation, letzter Zugriff: 04.06.2012

[url13] Wikipedia, "Model View ViewModel", Mai 2012, http://en.wikipedia.org/wiki/MVVM, letzter Zugriff: 04.06.2012

[url14] Wikipedia, "Web Services Description Language", Mai 2012, http://en.wikipedia.org/wiki/WSDL, letzter Zugriff: 04.06.2012

[url15] Wikipedia, "SOAP", Juni 2012, http://en.wikipedia.org/wiki/SOAP, letzter Zugriff: 04.06.2012

[url16] Wikipedia, "Representational state transfer", Juni 2012, http://en.wikipedia.org/wiki/REST, letzter Zugriff: 04.06.2012

[url17] Wikipedia, "JSON", Juni 2012, http://en.wikipedia.org/wiki/JSON, letzter Zugriff: 04.06.2012

[url18] System.Data.SQLite.org, "System.Data.SQLite: About", Juni 2012, http://system.data.sqlite.org/index.html/doc/trunk/www/index.wiki, letzter Zugriff: 05.06.2012

[url19] Doxygen.org, "Doxygen", Mai 2012, http://www.doxygen.org, letzter Zugriff: 08.06.2012

[url20] Microsoft Corporation, "C# Coding Conventions (C# Programming Guide)", 2012, http://msdn.microsoft.com/en-us/library/ff926074.aspx, letzter Zugriff: 05.06.2012

[url21] Doxygen.org, "Documenting the Code", Mai 2012, , http://www.stack.nl/~dimitri/doxygen/docblocks.html, letzter Zugriff: 06.06.2012

[url22] Doxygen.org, "Commands", Mai 2012, http://www.stack.nl/~dimitri/doxygen/commands.html, letzter Zugriff: 06.06.2012

EistCockpit

Anhang B

Vereinbarungen

David Schöttl, Remo Waltenspül & Diego Steiner

Aufgabenstellung Studienarbeit Abteilung I, FS 2012
Diego Steiner, Dave Schöttl, Remo Waltenspül

EistCockpit

1. Auftraggeber und Betreuer

Praxispartner und informeller Auftraggeber dieser Studienarbeit ist das Richtstrahl Bataillon 20 der Schweizer Armee.

Ansprechpartner Auftraggeber:

Major Hans-Andrea Veraguth hans-andrea.veraguth@gruenenfelder.ch 081/650 30 65
Hauptmann Michael Klötzli michael@kloetzli.li 077/422 03 41

Betreuer HSR:

Hauptbetreuung und fachliche Unterstützung: Thomas Corbat, Assistent IFS, tcorbat@hsr.ch
Benotung und Verantwortung: Prof. Dr. Markus Stolze, mstolze@hsr.ch

2. Ausgangslage

Innerhalb des Stabs eines Richtstrahlbataillons (Ristl Bat) in der Schweizer Armee existiert ein Teilbereich, die Einsatzstelle Telematik (Eist Tm). Deren Verantwortung ist die Planung und das Management des integrierten militärischen Fernmeldesystem-Netzes (IMFS), welches aus Richstrahverbindungen (Ristl Vrb), Richstrahlknoten (Ristl Kn) und diversen Funkgeräten besteht. Ein wichtiger Punkt dieser Arbeit besteht darin, einen genauen Überblick über den Status und die Disponibilität der einzelnen Standorte und Geräte zu verfügen. Die Eist Tm ist zudem die Anlaufstelle für die Ristl Stao bezüglich aller Belange (dies beinhaltet technische Probleme, Versorgungsanfragen, Planungsänderungen, etc.). Von den einzelnen Standorten empfängt die Eist Tm regelmässig Statusmeldungen.

Die Verwaltung all dieser Informationen basiert momentan hauptsächlich auf Meldezetteln in Papierform. Die Effizienz diese Systems ist mangelhaft und ermöglicht kaum, einen genügenden Überblick über die nötigen Informationen zu erlangen. Bisherige Ansätze zur Digitalisierung der Informationen (führen von Excel-Listen) haben nicht oder nur schlecht funktioniert.

Die Entwicklung und der testweise Einsatz im Ristl Bat 20 eines Systems, welches diese Missstände beheben soll, wird auch von höherer Eben unterstützt (Führungsunsterstützungsbrigade 41 - FU Br 41).

3. Ziele der Arbeit

Während dieser Studienarbeit wird ein verteiltes Informations-/Status-/Meldesystem entwickelt. Das entwickelte System soll im Stab eines Ristl Bat

eingesetzt werden können und die Mängel in den vorhandenen Arbeitsabläufen eliminieren.

Die genauen Anforderungen an ein solches System erarbeiten die Studenten während einem vierwöchigen Einsatz zu Beginn der Studienarbeit vor Ort im Stab des Rist Bat 20.

Als Auflage seitens der Schweizer Armee gilt lediglich, dass das System mit der bestehenden Infrastruktur ohne zusätzliche Hardware betrieben werden kann.

4. Durchführung

Mit dem HSR-Betreuer (Thomas Corbat) finden in der Regel wöchentliche Besprechungen statt. Zusätzliche Besprechungen sind nach Bedarf durch die Studierenden zu veranlassen.

Alle Besprechungen sind von den Studenten mit einer Traktandenliste vorzubereiten und die Ergebnisse in einem Protokoll zu dokumentieren, welches über das Projekt-Repository stets zugreifbar ist.

Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. An Meilensteinen (gemäss Projektplan) sind einzelne Arbeitsresultate in vorläufigen Versionen abzugeben. Über die abgegebenen Arbeitsresultate erhalten die Studierenden ein Feedback. Eine definitive Beurteilung erfolgt aufgrund der am Abgabetermin abgelieferten Dokumentation. Die Evaluation erfolgt aufgrund des separat abgegebenen Kriterienkatalogs in Übereinstimmung mit den Kriterien zur SA Beurteilung. Es sollten hierbei auch die Hinweise aus dem abgegebenen Dokument „Tipps für die Strukturierung und Planung von Studien-, Diplom- und Bachelorarbeiten" beachtet werden.

5. Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen. Die zu erstellenden Dokumente sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollen den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Die Dokumentation ist vollständig auf CD/DVD in 2 Exemplaren abzugeben.

Zudem ist eine kurze Projektresultatdokumentation im öffentlichen Wiki von Prof. M. Stolze zu erstellen.

6. Weitere Regeln und Termine

Im weiteren gelten die allgemeinen Regeln zu Bachelor und Studienarbeiten „Abläufe und Regelungen Studien□ und Bachelorarbeiten im Studiengang Informatik“ (https://www.hsr.ch/Ablaeufe-und-Regelungen-Studie.7479.0.html)

Der Terminplan ist hier ersichtlich (HSR Intranet)
https://www.hsr.ch/Termine-Diplom-Bachelor-und.5142.0.html

7. Beurteilung

Eine erfolgreiche SA zählt 8 ECTS-Punkte pro Studierenden. Für 1 ECTS Punkt ist eine Arbeitsleistung von ca. 25 bis 30 Stunden budgetiert. Dies entspricht ungefähr 17h pro Woche (auf 14 Wochen) und damit ca. 2 Tage Arbeit pro Woche.

Für die Beurteilung ist der HSR-Betreuer verantwortlich.

Die Bewertung der Arbeit erfolgt entsprechend der verteilten Kriterien-Liste.

Rapperswil, den 20. Februar 2012

Prof. Dr. Markus Stolze
Institut für Software
Hochschule für Technik Rapperswil

Geheimhaltungsvereinbarung für Studienarbeit "Eist Cockpit"

Kontext

Im Rahmen einer Studienarbeit entwickeln die Studenten der HSR David Schöttl, Diego Steiner und Remo Waltenspül ein Software System welches in Richtstrahl Bataillonen der Schweizer Armee eingesetzt werden kann. Zur Entwicklung dieser Arbeit haben sie Einblick in den Stabsbetrieb des Richtstrahl Bataillons 20. Dies beinhaltet auch den Kontakt mit Daten, welche nicht öffentlich zugänglich sind. Die drei Studenten, sowie der Betreuer Thomas Corbat, sind Angehörige der Armee. Entsprechend wird für die Durchführung der Arbeit keine Information benötig, welche für die Beteiligten nicht bereits im Rahmen ihrer Diensttätigkeit in der Armee zugänglich ist.

Auflagen

Durch diese Studienarbeit und alle dafür erstellten Artefakte dürfen keine schützenswerten Informationen (Stufen "Intern", "Vertraulich" und "Geheim"[1]) Unberechtigten zugänglich gemacht werden. Dies bedeutet:

- Die Handhabung von Information der genannten Klassifizierung sind zu berücksichtigen.
- Reale Daten mit der entsprechenden Klassifikation dürfen nicht in Datenbanken abgelegt werden. Für Testzwecke sind fiktive Daten zu verwenden.
- Die Projekt-Dokumentation enthält ebenfalls keine Informationen der genannten Klassifizierung. Sollte es für die Bewertung und das Verständnis zwingend notwendig sein solche Inhalte zu dokumentieren, hat dies in einem separaten Dokument zu erfolgen, welches nur berechtigte Personen einsehen dürfen. In der Projektdokumentation ist dies entsprechend zu referenzieren.
- Im Zweifel entscheidet der Stellvertretende Bataillonskommandant des Richtstrahlbataillons 20, Major Hans-Andrea Veraguth, wie im spezifischen Fall mit Informationen umzugehen ist.
- Alle Projektbeteiligten verpflichten sich, im Rahmen dieser Arbeit erlangte Kenntnisse über Personen oder System der Schweizer Armee vertraulich zu behandeln und nicht an dritte weiterzugeben.

Involvierte Personen

Name	Funktion	Datum, Ort	Unterschrift
Corbat, Thomas	Betreuer	08.06.2012, Glattbrugg	[signature]
Schöttl, David	Student	31/5/2012, Rapperswil	[signature]
Steiner, Diego	Student	31.05.2012, Rapperswil	[signature]
Stolze, Markus	Verantwortlicher	6.06.2012 Rappers...	[signature]
Waltenspül, Remo	Student	31.05.12	[signature]
Veraguth, Hans-Andrea	Bat Kdt Stv Rist Bat 20	i.V. 5.3.2012	Maj H.A.Veraguth Kdt Stv Ristl Bat 20 [signature]

[1] Verordnung über den Schutz von Informationen des Bundes http://www.admin.ch/ch/d/sr/5/510.411.de.pdf



Vereinbarung über Urheber- und Nutzungsrechte

Vereinbarung

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Projektarbeit *EistCockpit* von *David Schöttl, Diego Steiner & Remo Waltenspül* unter der Betreuung von *Markus Stolze* geregelt.

2. Urheberrecht

Die Urheberrechte stehen den Studenten zu.

3. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von den Studenten, von der HSR wie von der *Schweizer Armee* nach Abschluss der Arbeit verwendet und weiter entwickelt werden.

Rapperswil, den 6/6/2012 ....
Student: David Schöttl

Rapperswil, den 6. Juni '12 ....
Student: Diego Steiner

Rapperswil, den 06.06.2012 ....
Student: Remo Waltenspül

Rapperswil, den 6.6.2012 ....
Der Betreuer der Projektarbeit

Rapperswil, den 6.6.2012 ....
Der Studiengangleiter

Composing Integrated Systems Using GUI-Based Applications And Web Services

Mark Grechanik and Kevin M. Conroy
Systems Integration Group, Accenture Technology Labs
Chicago, IL 60601
Email: {mark.grechanik, kevin.m.conroy}@accenture.com

Abstract— **Integrated systems are composed of components that exchange information (i.e., interoperate [5]). These components include *Graphical User Interface (GUI) APplications (GAPs)* and web services. It is difficult to make GAPs interoperate, especially if they are closed and monolithic. Unlike GAPs, web services are applications that are designed to interoperate over the Internet.**

We propose a novel generic approach for creating integrated systems by composing GAPs with each other and web services efficiently and non-invasively. This approach combines a nonstandard use of accessibility technologies for accessing and controlling GAPs in a uniform way with a visualization mechanism that enables nonprogrammers to compose integrated systems by performing point-and-click, drag-and-drop operations on GAPs and web services. We built a tool based on our approach, and using this tool we created an integrated application that controls two closed and monolithic commercial GAPs and third-party web services. Our evaluation suggests that our approach is effective, and it can be used to create nontrivial integrated systems by composing GAPs with each other and web services.

I. Introduction

Integrated applications consist of components that exchange information (i.e., to interoperate [5]). Building integrated applications is difficult and expensive, since in addition to building components of these applications programmers should define protocols and implement the functionality for data exchanges between these components.

Components of integrated applications include *Graphical User Interface (GUI) APplications (GAPs)* and web services. Organizations use legacy GAPs to assist business operations. However, it is difficult to interoperate GAPs because many of them are closed and monolithic, and they do not expose any programming interfaces or data in known formats. Thus, while it is desireable to use GAPs as components in integrated applications, it is often difficult to add functionality to GAPs to enable them to interoperate with other applications.

Web services are software components that interoperate over the Internet, and they gain widespread acceptance partly because of the business demand for applications to exchange information [9]. Unlike GAPs, using web services enables organizations to quickly build integrated systems by composing these services for information exchange. By composing legacy GAPs with each other and web services into integrated systems, organizations can support their business processes better with these systems [8].

Integrating legacy GAPs with each other and web services is important for most organizations that have invested heavily in a variety of GAPs from multiple vendors [1]. Changing source code of GAPs to make them interoperable is difficult because of brittle legacy architectures, poor documentation, significant programming effort, and subsequently, the large cost of these projects. Modifying third-party GAPs may not even be possible since organizations often do not have access to the source code. Given the complexity of GAPs and the cost of making them interoperable, a fundamental problem of interoperability is how to build integrated systems by composing GAPs with each other and web services efficiently and non-invasively.

Our main contribution is a novel generic approach for composing GAPs with each other and web services into integrated systems. This approach combines a nonstandard use of accessibility technologies for composing GAPs written in different languages and running on different platforms in a uniform way with a visualization mechanism that enables nonprogrammers to compose integrated systems by performing point-and-click, drag-and-drop operations against closed and monolithic GAPs and web services. Since accessibility technologies are present on major computing platforms to allow disabled users to access applications, we utilize these technologies in our uniform mechanism of composing GAPs with each other.

We built a tool based on our approach, and we used this tool to compose two closed and monolithic commercial GAPs with web services into an integrated system. We describe our experience with this tool, and measure and analyze performance characteristics of the created integrated systems. The results suggest that our approach is efficient and effective.

II. A Motivating Example

E-procurement systems (EPS) are critical since they influence all areas of the organization performance [7]. Businesses employ elaborate EPSes that often consist of different GAPs assisting different steps of the purchasing process. In EPSes, the *rule of separation of duty* requires that operations be separated into different steps that must be done by independent persons (agents) in order to maintain integrity. With the separation of duty rule in place, no person can cause a problem that will go unnoticed, since a person who creates or certifies a transaction may not execute it. Implementing this rule results

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE



in agents using different GAPs that provide different services for different steps of the purchasing process.

Consider a typical e-procurement scenario. Employees order items using an electronic shopping cart service of the web-based application *OfficeDepot (OD)*. Department managers review selected items in the shopping cart, approve and order them, and enter the ordered items into *Quicken Expensable 98 (QE)*, a third-party closed and monolithic Windows GAP that the company uses internally to keep track of purchases.

The OD service sends a notification to a company accountant, who uses a closed and monolithic GAP called *ProVenture Invoices and Estimates (PIE)* to create invoices for ordered goods. When the ordered goods are received from OD, a receiving agent compares them with the entries in QE. The accountant can view but cannot modify records in QE, and likewise, no other agent but the accountant can insert and modify data in PIE. If the received goods correspond to the records in QE, the receiving agent marks the entries for the received goods in QE and notifies the accountant. After comparing the invoices in PIE with the marked entries in QE and determining that they match, the accountant authorizes payments.

In this example, each procurement agent uses different GAPs to accomplish different goals. Sometimes, several GAPs should be used to accomplish a single goal, and agents have to transfer data between these GAPs and perform other operations manually. Clearly, automating these activities is important in order to improve the quality and the efficiency of business services.

Consolidating disparate components into integrated EPSes enables enterprises to achieve a high degree of automation of their purchasing processes [8]. One supporting function of an integrated system is to extract information about ordered items from the service OD and create and enter invoices into PIE using this information. Once the payments are processed, the user marks invoices in PIE as paid, and the information about these invoices should be extracted from PIE and entered as expenses into QE. There are many other functions of the integrated EPS that involve interoperating GAPs with each other and web services.

An important function of an integrated EPS is to extend the functionalities of GAPs with web services. It means that users can continue to work with legacy GAPs, however, certain new functions will be delegated to web services. For example, when users modify expenses using QE, they may be required to submit the information about modified expenses to a web service that verifies these expenses using some business rules. These and many other functions reflect the need to compose GAPs with each other and web services in integrated systems.

III. How Our Approach Works

We present a birds-eye view of how our approach works by giving examples of composing GAPs with each other and web services to create integrated systems. First, we describe how to compose a simple integrated EPS from QE, PIE, and a web service, and then we show how to extend the functionality of QE with a web service.

A. Composing GAPs with Each Other And a Web Service

We have built a tool called *COmposer of INtegrated Systems (Coins)* that enables users to create intergrated systems by composing GAPs with each other and web services. Coins outputs integrated systems as composite web services that control and manipulate GAPs and other web services thereby making these components interoperate with one another. Specifically, we show how to compose an integrated system that extracts information about ordered items from the service OD and creates and enters invoices into PIE using this information. After these invoices are paid, the information about them is extracted from PIE and entered as expenses into QE.

The front end of Coins is shown in Figure 1. Using Coins, a user enters the name of the composite web service and the name of the exported method of this service (the defaults are `Service1` and `operation1`). The user also specifies that the service controls GAPs QE and PIE and the web service OD by providing information about them (i.e., their locations and *Web Service Definition Language (WSDL)* data). This information is shown in the leftmost tab (i.e., `Service Explorer`) of Coins.

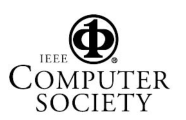
Next, the user enter an expense and an invoice into the GAPs QE and PIE correspondingly. Specifically, the user selects an expense envelope on the first screen of QE, and then double clicks on the entry in the envelope list box. These actions cause QE to switch to the expense entry screen. This and other screens of the GAP are captured and shown in the rightmost tab (i.e., `Screen View`) of Coins.

The purpose of interacting with GAPs is to allow Coins to record the structures of the screens and user actions on the GAPs and then transcode these actions into programming instructions that the resulting composite web service will execute. To do that, Coins intercepts selected user-level events using the *accessibility layer*[1]. These events allow Coins to record the sequence of screens that the user goes through as well as the actions that the user performs on GUI elements.

When recording the sequence of screens, Coins obtains information about the structure of the GUI and all properties of individual elements using the accessibility-enabled interfaces. This information allows the composite web service to locate GUI elements in order to set or retrieve their values or to perform actions on them in response to requests from its clients.

At the design time, the user should specify how to exchange data between components. Specifically, the user determines what GUI elements will receive the values of the properties of the web service OD and the values of GUI elements of other GAPs. For example, the value of the property `Item` of the service OD should be put into the GUI element labeled `Description` of the invoice screen of PIE, which in turn should be put into the GUI element labeled `(optional)` of the expense screen of QE.

[1] We discuss accessibility technologies in Section IV-A.



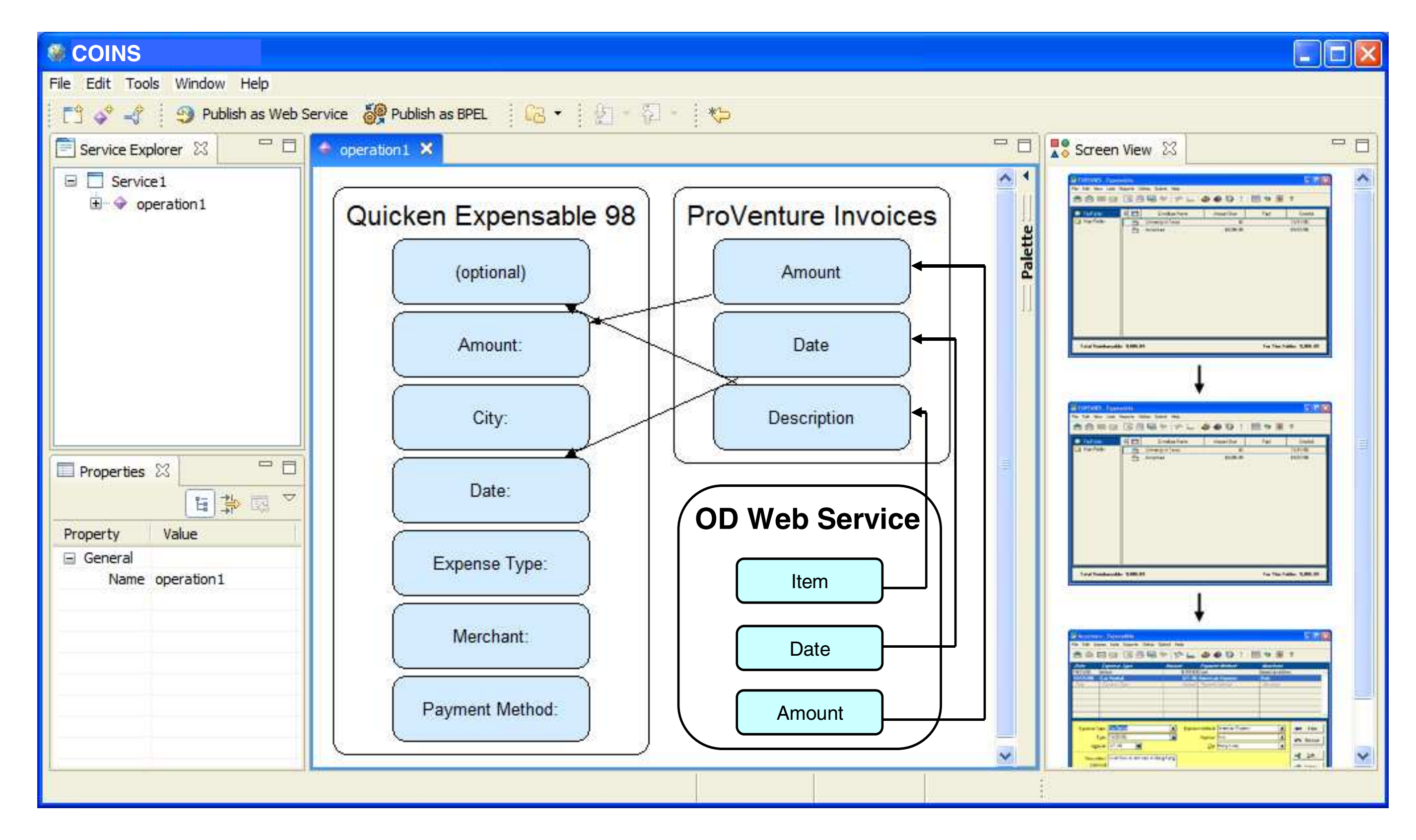


Fig. 1. The front-end of Coins.

To create mappings between these properties and GUI elements, the user moves the cursor over some GUI element of a GAP, and Coins uses the accessibility *Application Programming Interface (API)* calls to obtain information about this element. To confirm the selection, a frame is drawn around the element with the tooltip window displaying the information about the selected element. Then, the user clicks the mouse button and drags this element (or rather its image) onto the Coins' middle tab labeled with the name of the exported method (i.e., `operation1`) of the composite web service. After releasing the mouse button, the dragged element is dropped onto the Coins' dataflow palette under the label of the corresponding component.

Once the user has dragged-and-dropped all required GUI elements and loaded the description of web services from their WSDL files, it is time to connect these elements and properties of the loaded services with arrows that specify the directions of data exchanges. For example, by drawing an arrow between the element `Description` of the PIE and the element `(optional)` of QE, the user specifies that the data from the corresponding GUI element of PIE will be transfered to the GUI element of QE. While it is possible to specify how to transform the data during exchange, we do not consider these modifications in this paper for simplicity.

In addition, the user specifies what action(s) should be performed on GUI elements and what methods of the web services to call to initiate data exchange. For example, clicking on the tab `Invoice` in PIE initiates the procedure for extracting invoices. These actions are recorded as part of the workflow which can be exported as, for example a *Business Process Execution Language (BPEL)* program.

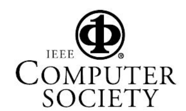
The resulting composite service is published by clicking on the button `Publish as Web Service` on the Coins' toolbar. Coins uses the information captured for each screen and input elements to generate Java code for the composite web service and deploy it to a web services platform such as Apache Axis. When this service is called from a client, the method `operation1` uses the accessibility interfaces to control and manipulate the GAPs and the web service to exchange information. A short movie demonstrating how Coins works is available at our website [www.markgrechanik.com]. It can be viewed inside the browser [http://www.markgrechanik.com/Coins.html] or downloaded and played as an AVI file [http://www.markgrechanik.com/Coins.avi].

B. Extending GAPs With Web Services

Legacy GAPs may be required to support new business processes. For example, a new business procedure may require that users submit the information about entered and modified expenses to a web service that verifies these expenses using some business rules before saving these expenses in QE. Since QE has been used for many years, integrating it with the new service allows the business to achieve new functionality at a low cost.

Coins allows users to extend the functionality of GAPs by integrating them with web services. The user connects GUI elements of QE with properties of the web service in the middle pane of Coins with arrows from the palette toolbox

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE



thereby specifying how data is transferred from the GAP to the service. Then, the user selects a method of the web service and determines how to invoke it. This method is invoked when a user performs some action on a GUI element (e.g., clicks a button). The values of GUI elements are passed as parameters to the invoked method, or they are used to set properties of the web service before the method is invoked.

In addition, the user specifies how to use the return values of the invoked method. They can be put in the selected GUI elements of the GAPs, or they can be displayed in message dialogs. The user can select an action in response to certain return values of the invoked method. Examples of these actions are terminating the GAP or going to the previous screen of the GAP whenever the latter action is possible.

IV. Elements of Our Solution

A key solution is to use GAPs as programming objects and GUI elements of these GAPs as fields of these objects, and to perform actions on these GUI elements by invoking methods on the objects that represent these GAPs. Unfortunately, services cannot access and manipulate GUI elements of GAPs as pure programming objects because GUI elements only support user-level interactions. Accessibility technologies overcome this limitation by exposing a special interface whose methods can be invoked and the values of whose fields can be set and retrieved thereby controlling GUI elements that have this interface. We give an overview of the accessibility technologies in the next Section IV-A.

A. Accessibility Technologies

Accessibility technologies provide different aids to disabled computer users [4]. Specific aids include screen readers for the visually impaired, visual indicators or captions for users with hearing loss, and software to compensate for motion disabilities. Most computing platforms include accessibility technologies since electronic and information technology products and services are required to meet the *Electronic and Information Accessibility Standards* [4]. For example, *Microsoft Active Accessibility (MSAA)* technology is designed to improve the way accessibility aids work with applications running on Windows, and *Sun Microsystems Accessibility* technology assists disabled users who run software on top of *Java Virtual Machine (JVM)*. Accessibility technologies are incorporated into these and other computing platforms as well as libraries and applications in order to expose information about user interface elements.

Accessibility technologies provide a wealth of sophisticated services required to retrieve attributes of GUI elements, set and retrieve their values, and generate and intercept different events. In this paper, we use MSAA for Windows, however, using a different accessibility technology will yield similar results. Even though there is no standard for accessibility API calls, different technologies offer similar API calls, suggesting slow convergence towards a common programming standard for accessibility technologies.

The main idea of most implementations of accessibility technologies is that GUI elements expose a well-known interface that exports methods for accessing and manipulating the properties and the behavior of these elements. For example, a Windows GUI element should implement the `IAccessible` interface in order to be accessed and controlled using the MSAA API calls. Programmers may write code to access and control GUI elements of GAPs as if these elements were standard programming objects.

B. Hooks

Hooks are user-defined libraries that contain *callback functions* (or simply *callbacks*), which are written in accordance with certain rules dictated by accessibility technologies. Hooks are important for Coins because they enable users to extend the functionality of GAPs, specifically to integrate them with web services without changing GAPs' source code. Writing hooks does not require any knowledge about the source code of GAPs.

In our approach, a hook library is generic for all GAPs, and its goal is to listen to events generated by the GAP into which this hook is injected as well as to execute instructions received from integrated systems. An example of an instruction is to disable a button until certain event occurs. The power of hook libraries is in changing the functionalities of existing GAPs without modifying their source code.

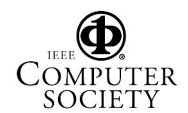
Main functions of the generic hook are to receive commands to perform actions on GUI elements, to report events that occur within GAPs, and to invoke predefined functions in response to certain commands and events. Since accessibility layers are supported by their respective vendors and hooks are technical instruments which are parts of accessiblity layers, using hooks is legitimate and accepted to control and manipulate GAPs. In addition, writing and using hooks is easy since programmers use high-level accessibility API calls, and they do not have to deal with the complexity of low-level binary rewriting techniques.

When a target GAP is started, the accessibility layer loads predefined hook libraries in the process space of this applications and registers addresses of callbacks that should be invoked in response to specified events. Since hooks "live" in the process spaces of GAPs, their callbacks can affect every aspect of execution of these GAPs.

V. Architecture

The architecture of Coins is shown in Figure 2. Solid arrows show command and data flows between components, and numbers in circles indicate the sequence of operations in the workflow.

This choice of the architecture is influenced by the fact that in enterprise environments GAPs and web services are often located on different computers. Some GAPs are located on the same computer, but they may not be started at the same time due to certain constraints. For example, two instances of the same application cannot bind their sockets to the same port on the same computer, and subsequently, they cannot



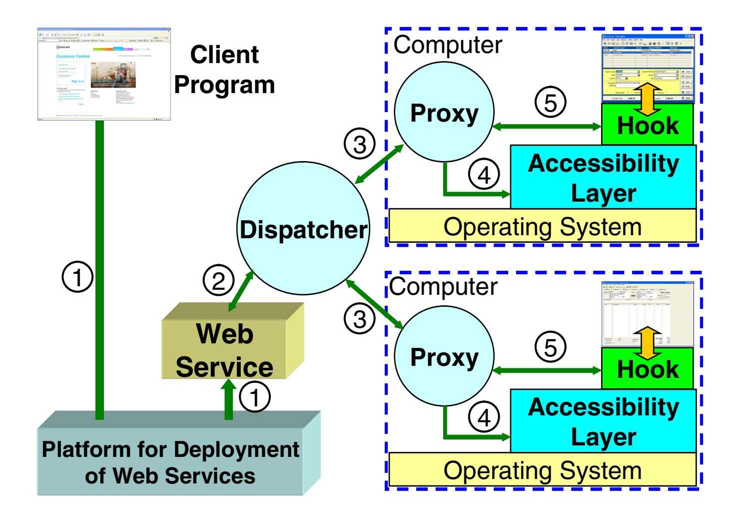


Fig. 2. The architecture of Coins.

run simultaneously. This and some other conditions force administrators to distribute GAPs and web services across computers in enterprise environments.

Accessibility technologies cannot control distributed applications. If an integrated system (e.g., a composite web service) and the GAPs it controls are located on different computers, then the service cannot use any accessibility technology to control these GAPs. A solution to this problem is to use proxies to control GAPs by sending commands to them from integrated systems. A Proxy is a generic program that receives requests from integrated systems, extracts data from GAPs in response to these requests, and sends the extracted data back to these integrated systems. Proxies use the accessibility layer to control and manipulate GAPs.

The Dispatcher is the central point for coordinating proxies in the distributed environment. It is a daemon program that collects information from proxies and makes decisions to which proxy to forward requests from composite web services that represent integrated systems. For example, if copies of the same GAP are installed on different computers, the Dispatcher assign the instances of this GAP to different requesting clients thereby enabling their execution in parallel.

Since web services and GAPs may be moved around the enterprise computers for different reasons (e.g., to improve business processes or the performance of applications), the Dispatcher provides the migration and location transparencies for web services and GAPs. Proxies register with the Dispatcher under unique names, collect information about GAPs located on their computers, and send this information to the Dispatcher. The Dispatcher receives tables of GAPs from proxies on a regular basis, and it uses this information to direct requests from web services to appropriate GAPs.

When a method of the composite web service, which represents the integrated system is invoked for the first time by its client `(1)`, the service connects to the Dispatcher and sends a registration request `(2)`. From this request the Dispatcher determines what GAPs are required to run the web service. The Dispatcher looks up the GAP tables received from connected proxies, and once it finds the required GAPs, it sends requests to the corresponding proxies to reserve these GAPs for the web service `(3)`.

After the Proxy starts the GAP, it uses the accessibility layer to inject the hook library into the GAP `(4)`. The hook spawns a thread within the GAP's process in order to establish a communication channel with the corresponding Proxy. Using this channel, the Proxy can send commands and receive notifications of the events that occur within the GAP and for which callbacks from the hook are registered `(5)`. Thus the generic hook can be viewed as a virtual machine that can be used to control and manipulate GAPs.

VI. Code Generation

Coins generates code for the resulting integrated system (i.e., a composite web service). The structure of the code reflects dependencies between GUI elements in GAPs. In event-based windowing systems (e.g., Windows), each GAP has a main window (which may be invisible), which is associated with the event processing loop. Closing this window causes the application to exit by terminating the loop. The main window contains other GUI elements of the GAP. A GAP can be represented as a tree, where nodes are GUI elements and edges specify children elements that are contained inside their parents. The root of the tree is the main window, the nodes are container elements, and the leaves of the tree are basic elements (e.g., buttons or edit boxes).

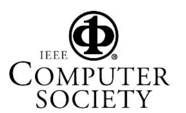
GAPs are state machines whose states are defined as collections of GUI elements, their properties (e.g., style, read-only status, etc.), and their values. When users perform actions they change the state of the GAP. In a new state, GUI elements may remain the same, but their values and some of their properties change.

Coins takes inputs describing the states of the GAPs and generates classes whose methods control GAPs by setting and getting values of their GUI elements and causing actions that enable GAPs to switch to different states. When the user switches the GAP to some state, Coins records this state by traversing the GUI tree of the GAP post-order using the accessibility technology. For each node of the tree (i.e., a GUI element), Coins emits code for classes are linked to these GUI elements, and these classes contain methods for setting and getting values and performing actions on these elements. Coins also emits the code that handles exceptions that may be thrown when web services control GAPs.

VII. Prototype Implementation

We implemented Coins in Windows using C++ and Java. The prototype implementation is based on the MSAA toolkit version 2.0 and an MS XML parser, as all communications between the components of Coins are in XML format. Our prototype implementation included the front end, the Dispatcher, the composite service code generator, the Proxy, and the hook. We wrote the front end and the generator in Java and the rest of components of Coins in C++. We used sockets as an

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE



interprocess communication mechanism. For our prototype we used Apache Axis 2, which is a platform for development and deployment of web services [http://ws.apache.org/axis2]. For a full-scale deployment of web services using the Web Services Deployment Descriptor under Axis we refer the reader to the Axis documentation [2]. Our implementation contains close to 12,800 lines of code.

VIII. Experimental Evaluation

In this section we describe the methodology and provide the results of experimental evaluation of Coins. We describe case studies in which we successfully integrate two commercial closed and monolithic GAPs and a web service in an instance of the EPS which is described in Section II; we demonstrate how batch data exchanges result in significant performance gain using our approach; and we conduct experiments to analyze the performance penalty when using GUIs to access GAP services in integrated systems versus invoking methods of pure programmatic web services. We show how to use our approach so that performance penalty incurred by communicating with GAPs through their GUI elements stays below some acceptable limit.

A. Case Study

To demonstate our approach, we created an E-procurement system (EPS) from QE and PIE, as we described in Section II. Recall that QE and PIE are closed and monolithic commercial GAPs that run on Windows. QE allows users to enter and track expenses, and PIE allows users to create and print invoices, estimates, and statements, and to track customer payments and unpaid invoices.

Our experience confirms the benefits of our approach. We created an integrated EPS from both QE and PIE and a web service without writing any additional code, modifying the applications, or accessing their proprietary data stores (see Section III). We carried out experiments using Windows XP Pro that ran on a computer with Intel Pentium IV 3.2GHz CPU and 2GB of RAM.

In our case study, we compared the effort required to create an integrated EPS using Coins with the programming effort to create the same service by using the source code of GAPs. We created applications similar in functionalities to QE and PIE respectively. It took us approximately fourteen hours to create and test our imitations of QE and PIE which we call IQE and IPIE respectively. Then, we created an integrated EPS. It took us approximately three and half hours to extract the code from the IQE and IPIE, move it to the integrated system project, and compile and debug it. Compared to that, it took us less than fifteen minutes to generate an integrated system using Coins.

B. Performance Considerations

Calling methods directly is more efficient than invoking services of GAPs through their GUI elements. The additional overhead cost, OC, consists of the GAP startup time, the initialization time for the internal structures representing GUI elements, screen switching time, and communicating time between Proxies and GAPs. Common delay, CD, for both programmatic and GAP-based integrated systems consists of network latency time of transmitting method call requests between components and delivering results back and the method execution time. The GUI computation overhead (GCO) ratio in percent, $\text{GCO}=\frac{\text{OC}}{\text{CD}} \cdot 100$, shows what percentage of the execution time is dedicated to handling GAPs and their GUI elements.

C. Performance Evaluation

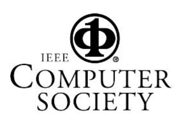
The goal of the performance experiment is to evaluate how much performance penalty GAP-based integrated incur versus pure programmatic ones. In addition, we show that using batch data exchanges results in significant performance gain.

1) Performance Stress Test: We designed the performance test to measure the reliability and sustainability of transaction processing throughput of our implementation of the EPS system. The test script simulated users who individually order 100 items from the web service OD. For each invoice entered in IPIE, the process included extracting this invoice and creating a corresponding expense in IQE, with the last step in this process being either the return to the initial screen of the GAP, or the termination of the GAPs. Each virtual user therefore completes 100 individual transactions during a user session. The test was run at a user load for duration of 24 hours in order to minimize various effects of other applications and services running on the same computer as well as network irregularities over the extended period of time.

We repeated this test four times for our implementations of IQE and IPIE to collect more data. The first test was run with the IQE and IPIE implemented as purely programmatic components (libraries) with no GUI interfaces used to transfer data. The second test was run against IQE and IPIE whose screens were reused by returning to original screens, not restarting the GAPs. The third test was run against the IQE and IPIE which were restarted after each transaction. The last test was run with IQE and IPIE whose GUIs contained different multimedia elements (e.g., animations and bitmaps). We report an average time per transaction for each test.

Experimental results from evaluating how much performance penalty these GAP-based data transfers incur versus pure programmatic ones are shown in Figure 3. The vertical axis shows the average time in seconds per data exchange transaction, and the bars correspond to the tests. The fastest transaction takes on average 0.8 seconds when no GAPs are used, that is the data is passed purely programmatically between IQE and IPIE libraries. The performance drops when GUI elements of IQE are used to encapsulate the functionality required to transfer the data. The average time per transaction increases to 2.2 from 0.8, which is 175% increase. The difference between these average transaction times is 1.4 second, which we attribute to the overhead of the GUI computations.

The situation worsens for the third test when the GAP is required to restart every time the data exchange is performed. The overhead associated with restarting of the GAP increases the average time per transaction to 4.6 seconds. Finally,



when the GAP uses multimedia images and animations, the performance becomes worse, taking on the average time per transaction 7.2 seconds.

2) Batch Data Exchanges: Coins enables users to specify batch data exchanges between components of integrated systems. Sending messages between components is delayed in batch message transfers until a batch message containing many smaller messages is formed and transmitted at once. The combined overhead associated with sending many small messages is higher than the overhead of sending one larger message that contains these small messages. Sending separate small messages involves adding control information in headers that specify additional information for message transfer, invoking functions that create and parse these messages, and often having to perform additional extraction operations on GUIs. Extracting all data from GUI elements and sending them to the destination in one single message enables users to amortize a one-time overhead over many data items in the message.

In addition, since the network latency time is one of the major contributors to the overhead, sending fewer messages may result in the reduction of this overhead. Specifically, sending one large message instead of many small messages reduces the communication overhead between the Dispatcher and Proxies. Also, when sending many messages, next message is usually send when the receipt of the previous message is acknowledged by a special response message. Sending and waiting for these acknowledgement messages adds unnecessary computation and communication overhead, which can be significantly reduced using batch transfers.

The goal our experiment is to show that exchanging messages in batches yields better performance of the integrated system. The graph showing the dependency of transfer time per item from the number of items transferred between GAPs is shown in Figure 4. The horizontal axis shows the number of data items transferred per transaction. An average size of the data item is approximately 360 bytes. The vertical axis shows the time it takes to transfer an item from IPIE to IQE. For a single item it takes approximately 2.1 seconds to transfer a data item considering the GAP computational overhead.

As data items are transferred in batches, the amount of time it takes to transfer an item drops approximately 14 times to close to 0.15 second as the number of the items in a batch grows to 500. Correspondingly, the size of the XML message containing these data items grows from 1.5Kb to 130Kb. However, transferring 1,000 items instead of 500 reduces the transfer time per item by only 1.9 times. Our explanation is that it takes longer to create and parse large messages as well as to transfer them, and this additional overhead dwarfs the reduction of the transfer time per message. Still, the our experiment shows that it is possible to gain significant performance by performing batch transfers between GAPs.

D. Recommendations

Choosing Coins versus writing programmatic integrated systems is a matter of trade-offs between the development effort and the resulting performance. If the performance of the integrated system is a critical issue, then developing a programmatic application is the right choice. However, minimizing cost is an important trade-off for performance, and using Coins allows users to reduce the development cost significantly while keeping performance overhead within acceptable limits.

Recall that the GUI overhead GCO is inversely proportional to the common delays CD, such as network latency and GAP backend computations. At one extreme, CD is much smaller than the OC, and the GCO is high. For example, if the OC is one second and the CD is one tenth of a second, then the GCO is 1,000%. Since the GCO is high, users should consider to develop a programmatic integrated system. At the other extreme, CD is much higher than the OC, and the GCO is small. For example, if the OC is one second and the CD is 20 seconds, then the GCO is 5%.

Based on our conversations with many professionals who build, deploy, and maintain integrated systems, these professionals are willing to consider up to 10% of performance penalty if they can reduce the development effort. We observed that for many commercial applications backend computations take from five to fifteen seconds. It means that for the GCO to be less than 10%, its absolute value should be between 0.5 to 1.5 seconds, which is consistent with the GCO of 1.4 second

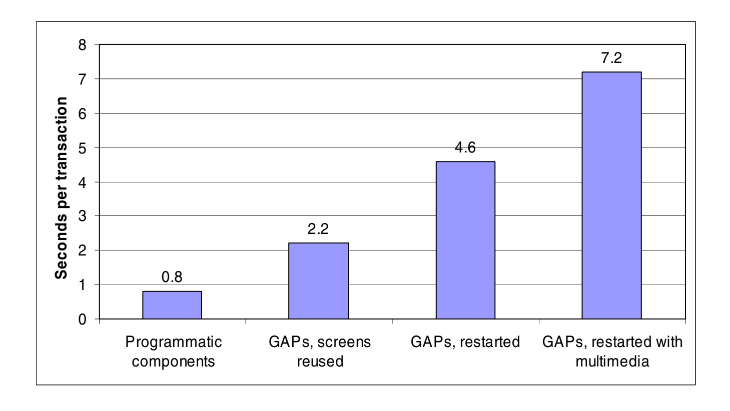


Fig. 3. Transaction throughput for web services.

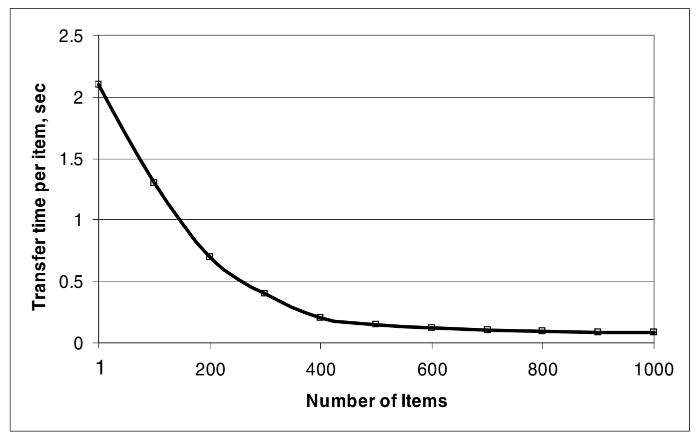
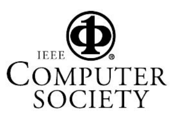


Fig. 4. Dependency of transfer time per item from the number of items transferred between GAPs using the batch exchange.

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE



which we measured in our performance experiment with the data transfer between IQE and IPIE.

Since GAPs consume significant CPU time for GUI painting when images and animations are included, using these applications for integrated systems may not be possible for performance reasons. In practice, clients use composite web services as integrated systems via the Internet, and these services use GAPs via the LAN. From this perspective the performance penalty incurred by using GAPs is minimal since the low-level communication mechanisms such as transmission, marshaling and unmarshaling network data have the largest overhead common to all solutions.

E. Limitations

In general, Coins may not work well with GAPs whose GUIs are frequently modified since it would require users to regenerate integrated systems to adjust for new GUIs. However, the number of such GAPs is small, and most GUIs are stable and may have small changes between releases, which happen infrequently. It is a bigger problem with web-based applications whose GUIs change relatively frequently.

In general, managing many instances of the same GAP is difficult. For example, when running web-based applications, they open many popup windows. These windows and processes that control them are not linked explicitly to the application that opened them. Thus, when two or more web-based applications ran on the same computer simultaneously, data from these applications may be mixed. We are currently working on solving this problem.

IX. Related Work

UniFrame is a framework for building integrated systems by assembling pre-developed heterogeneous and distributed software components [15]. The glue/wrapper code that realizes the interoperation among the distributed and heterogeneous software components can be generated from the a descriptive model. Unlike UniFrame, Coins does not require users to write code for models, and it does not require the knowledge of the source code of components.

A web browser-shell integrates a command interpreter into the browser's location box to automate HTML interfaces [13]. A browser-shell wraps legacy CLPs with an HTML/CGI graphical interface. This approach is heavily dependent upon parsing HTML and extracting data from the command line input/output, and in that way it is significantly different from our approach which does not need to parse any source code.

Code patching [6] and binary rewriting [11] techniques modify the binary code of executable programs in order to control and manipulate them when integrating these programs into composite systems [10]. However, these techniques are platform-dependent, and programmers are required to write complicated code to change program executables. Using these techniques is difficult and error prone, and often causes applications to become unstable and crash.

When it comes to extracting information from GAPs and their GUI elements, the term *screen-scraping* summarily describes various techniques for automating user interfaces [14] [3]. Macro recorders use this technique by recording the users mouse movements and keystrokes, then playing them back by inserting simulated mouse and keyboard events in the system queue [12]. Our approach differs fundamentally from other screen-scraping techniques since it modifies GAPs to extend their functionalities while, by definition screen scrapers only deliver information describing GUIs. In addition, Coins does not depend on parsing a scripting language that describes the GUI, and therefore it is more generic and uniform.

X. Conclusion

We proposed a novel generic approach for creating creating integrated systems by composing GAPs with each other and web services efficiently and non-invasively. This approach combines a nonstandard use of accessibility technologies for accessing and controlling GAPs in a uniform way with a visualization mechanism that enables nonprogrammers to create web services by performing point-and-click, drag-and-drop operations.

We built a tool based on our approach, and we used this tool to compose an integrated system from two closed and monolithic commercial GAPs and web services. Our case study showed that a key advantage of Coins is that it involves minimal development efforts. Our evaluation suggests that our approach is effective and it can be used to create integrated systems from nontrivial legacy GAPs and web services.

References

[1] Building software that is interoperable by design. *http://www.microsoft.com/mscorp/execmail/2005/02-03interoperability-print.asp*.

[2] Documentation for Apache Axis 1.2. *http://ws.apache.org/axis/java/index.html*.

[3] Screen-scraping entry in Wikipedia. *http://en.wikipedia.org/wiki/Screen_scraping*.

[4] Section 508 of the Rehabilitation Act. *http://www.access-board.gov/508.htm*.

[5] *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries.* Institute of Electrical and Electronics Engineers, January 1991.

[6] B. Buck and J. K. Hollingsworth. An API for runtime code patching. *Int. J. High Perform. Comput. Appl.*, 14(4):317–329, 2000.

[7] D. Burt, D. Dobler, and S. Starling. *World Class Supply Management: The Key to Supply Chain Management.* McGraw-Hill Irwin, July 2002.

[8] A. N. K. Chen and B. B. M. Shao. Web services enabled procurement in the extended enterprise: An architectural design and implementation. *J. Electron. Commerce Res.*, 4(4):140–155, 2003.

[9] C. Ferris and J. A. Farrell. What are web services? *Commun. ACM*, 46(6):31, 2003.

[10] M. Grechanik, D. S. Batory, and D. E. Perry. Integrating and reusing GUI-driven applications. In *ICSR*, pages 1–16, 2002.

[11] J. R. Larus and E. Schnarr. EEL: Machine-independent executable editing. In *PLDI*, pages 291–300, 1995.

[12] R. C. Miller. End-user programming for web users. In *End User Development Workshop, Conference on Human Factors in Computer Systems*, 2003.

[13] R. C. Miller and B. A. Myers. Integrating a command shell into a web browser. In *USENIX Annual Technical Conference, General Track*, pages 171–182, 2000.

[14] B. A. Myers. User interface software technology. *ACM Comput. Surv.*, 28(1):189–191, 1996.

[15] W. Zhao, B. R. Bryant, C. C. Burt, R. R. Raje, A. M. Olson, and M. Auguston. Automated glue/wrapper code generation in integration of distributed and heterogeneous software components. In *EDOC*, pages 275–285, 2004.

0-7695-2925-9/07 $25.00 © 2007 **IEEE**



An Optimistic Technique for Transactions Control using REST Architectural Style

Luiz Alexandre Hiane da Silva Maciel
Instituto Tecnológico de Aeronáutica (ITA)
Praça Marechal Eduardo Gomes, 50
Vila das Acácias, CEP 12228-900
São José dos Campos - SP - Brasil
luizhiane@gmail.com

Celso Massaki Hirata
Instituto Tecnológico de Aeronáutica (ITA)
Praça Marechal Eduardo Gomes, 50
Vila das Acácias, CEP 12228-900
São José dos Campos - SP - Brasil
hirata@ita.br

ABSTRACT

SOA is a service oriented architecture that allows development of software with interoperability and weak coupling. Nowadays WS-* is the most used SOAP-based specification set for constructing web services. REST is an architectural style that permits the development of services in a simpler way and obeys the SOA's paradigm, however, it does not provide standardized support to address some non-functional requirements of services, for instance, security, reliability, transaction control. This article proposes a technique, based on REST, to support the web services transactional control implementation. The technique uses the optimistic method to control distributed systems transactions. An example of application was implemented to show its feasibility.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures—*Patterns*; H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

DESIGN, EXPERIMENTATION

Keywords

Architectural style, concurrency control, REST, transaction, web services

1. INTRODUCTION

The Service Oriented Architecture (SOA) has been gaining attention due to the ability to build interoperable and loose coupled distributed applications. SOA enables software components reusability so that new applications can be developed by composing existing components. SOA promotes the interoperability between different platforms, systems and programming languages.

The two most adopted SOA implementations are: the WS-* and REST (Representational State Transfer). REST is both more recent and a "lighter" form to implement services than WS-*. WS-* is a set of specifications for development of services based on SOAP [14] and WSDL [15]. These specifications were developed jointly by several organizations such as BEA, IBM and Microsoft. Among the specifications are *WS-Security* [7], *WS-Reliability* [6], *WS-Transaction* [8], *WS-Coordination* [9].

In general, the WS-* specifications are designed to work together by using SOAP extensibility model. The specifications support the non-functional requirements implementation for applications that require a more complex infrastructure. In those applications aspects such as security and reliability are essential to meet the business goals.

SOAP currently is more used to implement remote procedure call (RPC). In SOAP, the HTTP protocol is used to transport an XML document which holds information such as method invoked, input parameters and output expected [10]. SOAP envelope is transported inside HTTP envelope.

REST is an architectural style for web services implementation. Services implementation that follows this architectural style makes extensive use of HTTP protocol characteristics. That form of implementation is called RESTful web services.

RESTful web services are simpler to understand and implement than those which adhere to the WS-*. However, many of the non-functional requirements in web services are not yet addressed, such as security, reliability and control of transactions. In the transaction control, in general, various resources have to be updated in a consistent way through distributed transactions. The goal is to ensure the ACID properties (atomicity, consistency, isolation, durability). However, not all of the properties can be ensured in web services context. For web services, there are alternatives to accomplish data consistency and integrity as the transaction compensation.

This work aims to provide a proposal to support transaction implementation in RESTful web services. The proposal uses non-lock concurrency control algorithms for REST architectural style. We try to provide transactional support without adding complexity on the simplicity provided by the Fielding proposal [2].

Sections of this work are organized as follows: Section 2 proposes the use of optimistic transaction control to address

SAC'09 March 8-12, 2009, Honolulu, Hawaii, U.S.A.

the need to execute transactions involving RESTful services. Section 3 brings an analysis of the proposal and indicates some good practices raised for usage. In Section 4, some conclusions and proposals for future work are presented.

2. OPTIMISTIC CONCURRENCY CONTROL FOR REST

This section presents a proposal to promote the transactional control for RESTful services through the use of non-lock concurrency control.

Some problems in the infrastructure necessary for web services implementation are not discussed, such as host failure, broken link, unavailability of the web server or database, since they are not central to understand our proposal. The fault-tolerance of the infrastructure is an important issue in the realm of web, and it will be addressed in future work.

2.1 REST – Representational State Transfer

REST intends to evoke the image of how a well designed web application behaves: a network of web pages (a virtual state machine) where user interacts with the application by selecting links (states transitions), resulting in the transfer next page to him, which is rendered for his use [2].

The Web is composed of resources, which may be considered items of interest. For example, suppose that customers of a bank wants to access information about their accounts. Thus, the bank can create a resource called *account/{id}*, where {*id*} should be replaced by the client account number. The URL for a customer with the account number 12345 would be: `http://www.bank.com/account/12345`

In asking that URL, a resource *representation* is returned, for example, *account12345.html*. The client application is placed in a *state*, which displays the information to the user through a web browser. The user can click on hyperlinks within account12345.html and access other resources available on the bank's server, as a page to make online payments or an resource that represents their savings account.

Thus, other representations are accessed, putting client application in a new state per request. User can continue interaction indefinitely. Therefore, the client application changes (*transfers*) its state for each resource representation received. Hence, emerges the definition of Representational State Transfer - REST.

According to Fielding [2], the REST key abstract information is a *resource*. Any information that may be appointed can be a resource: document, image, service, collection of other resources, non-virtual object and so on.

In SOA context, resource is a conceptual entity that identifies a service exposed to clients. It is usually a noun, since the verbs better indicate the action on the resource and not their identification.

2.1.1 Architectural Style

REST is an architectural style, it is not a standard. It does not make much sense to create a specification for REST because it is just a style that have to be understood to design web services in that style [1]. It is possible to make an analogy with the client-server architecture style, widely known. There is not a specification called client-server, but only rules that must be followed by applications that wish to follow that style.

An architectural style is a coordinated set of architectural constraints that restricts the roles/features of architectural elements and the allowed relationships among those elements within any architecture that conforms to that style.

According to Fielding [2], REST has a series of architectural restrictions that, when applied as a whole, emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.

Some of main features proposed by REST are: Client-Server paradigm – the important principle is the *separation of concerns* between the client and server, allowing them to evolve independently. Stateless – Communication between client and server must be stateless. The server does not store conversational state with clients. Session state is kept entirely on the client. Cache – the responses to prior requests can be reused in response to later requests that are equivalent and likely to result in a response identical to that in the cache if the request were to be forwarded to the server. Uniform Interface – A central feature of REST is the uniform interface between the components. The overall system architecture is simplified and there is an improvement in visibility of interactions. Layered System – System hierarchically organized in which each layer provides services for the top layer and uses those of the bottom layer. Code on demand – the client can extend its functionality by downloading and executing code from the server as he needs. Thus, the client is simplified since it reduces the number of features required to be pre-implemented.

While REST is not be a standard, RESTful services implementation uses standards, such as HTTP, URL, XML, HTML, GIF, JPEG. In general, Web can be considered as a REST system [1]. Many of the services used in the Web, as search services, booking-ordering services, dictionary services can be considered RESTful web services, at least partially.

Richardson and Ruby [10] define a new term to indicate a concrete architecture that implements the REST architectural style called the *Resource-Oriented Architecture - ROA*. Basically, it makes use of URI, HTTP and XML. ROA is an implementation of REST, which in turn complements SOA. Therefore, ROA is only a term coined by Richardson and Ruby to make clear the difference between REST and a concrete implementation of that architectural style.

According to them [10], in order to identify a resource, ROA uses *URI - Universal Resource Identifiers* [13]. URI is the name and the address of a resource. If an information does not have a URI, it is not a resource and may not be directly accessed on the web, because the URI is which allows the access of such information providing a single address. It is important that URIs have a clear correspondence with the resources they identify.

ROA proposes the use of HTTP fundamental methods for the most common operations to address the need for a uniform interface [10]. The common operations are referenced as CRUD (create, retrieve, update, delete). The fundamental HTTP methods include: HTTP GET retrieves information about a resource; HTTP PUT creates a new resource when new URI is provided by the client, i.e., the client defines the URI that will become available; HTTP POST creates a new resource without providing new URI, i.e., the service itself defines the URI that will become available; HTTP PUT updates an existing resource; and HTTP DELETE erases an existing resource.

2.1.2 Transactions as Resources

In ROA each HTTP request has one resource as a destination, but some services exposes operations that generate multiple resources, such as the operation of transferring funds from a checking account to a savings account.

In order to solve this problem, Richardson and Ruby [10] propose to expose the transactions as resources. The following steps are proposed:

1. Client creates a transaction by sending a POST to a transaction factory resource.
2. The response provides the URI for the created resource.
3. Client submits the first part of the transaction which updates the balance of the checking account by sending a HTTP message.
4. Client submits second part of the transaction which updates the balance of savings account by sending a HTTP message.
5. At any time it is possible to make a rollback by sending a DELETE to the transaction URI.
6. Client sends a request to commit the transaction.
7. The server must ensure that the transaction maintains the resource in a consistent state. If everything is successfully executed, the transaction is committed and the resources updated.

In the server, the procedure is implemented by a component that receives the various requests and creates a queue of actions associated with the transaction. When requested to commit, the server initiates a transaction in the database, applies the actions in the queue and then commits the transaction. In case of failure, it is propagated as a transaction commit failure [10].

2.2 Optimistic Concurrency Control

The optimistic method for concurrency control does not use locks to preserve the integrity and consistency of the application. It is called optimistic because it assumes that conflict between different transactions operations does not occur or rarely occurs. The transactional control is as permissive as possible.

Because it does not use locks, it is deadlocks free. Moreover, if the majority of transactions is just reading transactions the overhead for concurrency control becomes negligible.

The optimistic concurrency control can be summarized as follows [5]: (1) Reading of a value can not cause loss of integrity, so there is no restriction on several concurrent reading transactions. (2) Writes are severely restricted for concurrent transactions. Each transaction is composed of two or three phases: a read phase , a validation phase and a possible write phase. During the read phase, all writes take place on the local copies of data to be modified. Then, if it can be established during the validation phase that the changes the transaction made will not cause a loss of integrity, the local copies are made global in the write phase.

Conflicts are hoped to not occur in optimistic concurrency control [5]. Conflicts are the worst case in an optimistic approach, i.e., validation phase fails and the transaction is interrupted and start over again as a new transaction. Thus a transaction will have a write phase only if the preceding validation succeeds.

When conflicts are rare, the validation phase can be made efficiently, allowing a good performance of the transactions. But if conflicts are frequent, the cost of repeatedly transactions restart impacts negatively on performance.

2.3 REST with Optimistic Concurrency Control

The procedure exposed in Subsection 2.1.2 assumes that all resources participating in a transaction are present in the same server. On the server, the requests (HTTP PUT) are stored in a actions queue. When a client sends a commit request (HTTP PUT with *committed* = *true*), the server delegates transaction execution to a database. The server starts a database transaction, applies all queued actions and commits the database transaction.

If the transaction is distributed, i.e., if the operations are executed between different banks, with different URIs and databases system, the proposal does not work because it assumes that there is a single database responsible for executing the transaction. For example, assuming that there are two servers, each with its own database, and that the server holding the debit account is not operational, the transaction executes only the credit operation, which contradicts the atomicity property.

Therefore, the procedure of transactions in Subsection 2.1.2 does not support control of distributed transactions. The services operate only as a facade for the transaction operations implementation, i.e., they are only an interface to a database transaction.

It is possible to propose new ways to deal with distributed transactions in RESTful services. The proposal presented in this paper uses the optimistic concurrency control. To do so, each resource has to inform within its representation (for example HTML, XML or text) one more data item to indicate its current version. The value of the current version has to be increased every time the resource is updated. When a resource is created, an initial version value is assumed.

When two clients read a resource in the same version, the first to change the resource and request an update causes the resource update and version increment. When the second client requests an update, a conflict is identified, because the version he wants to change no longer exists. The transaction is aborted or a list with the conflicts is returned to the client to decide what should be the resource final version. What can be made when a conflict is identified is up to each application.

In order to demonstrate the feasibility of the proposed optimistic transactional control, an example was implemented for the transfer of funds between bank's accounts. Two clients try to transfer funds concurrently.

Figures 1, 2, 3, 4 illustrate the scenario. The filled arrows represent the requests made by clients. The dashed arrows represent the responses sent by servers to clients. The requests are numbered to indicate the order of execution and the responses indicate whether there is success or failure in processing the requests.

The goal of both clients is to transfer $50 from 12345 account to 98765 account. Both accounts have initial balance of $500.

The status code of HTTP 200 (OK) indicates that the

server successfully executed the action requested by the client. The status code HTTP 409 (Conflict) means that the client tried to put the resources in an inconsistent state.

The first step of each client is to get the representation of resources involved by sending an HTTP GET request to the banks. Both clients get the first version of the accounts 12345 and 98765. See Figure 1. The balance of both accounts is $500.

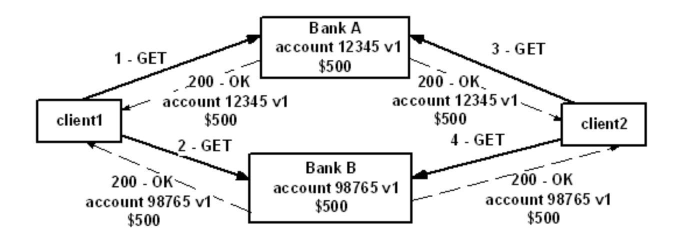


Figure 1: Each client retrieves the accounts representation.

A possible HTTP request message to retrieve account 12345 can be:

```
GET /accounts/account/12345 HTTP/1.1
Host: bank.com
```

Below it is illustrated a response to a request for the account 12345.

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: 142
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<account>
<accountNumber>12345</accountNumber>
<balance>500</balance>
<version>1</version>
</account>
```

With the representations, clients make fund transfers in their local accounts representation. At the time to commit the updates in the banks, the client 1 achieves the update of the account 12345, which goes to version 2 with balance $450. The client 2 achieves the update of the account 98765, which goes to version 2 with balance $550. When the client 1 tries to change the account 98765, a conflict is detected because there is a newer version of it. The account 98765 older version no longer exists. Similarly, the client 2 can not update the account 12345. See Figure 2. The contents of a request to change the balance of the 12345 account (version 1) to $450.00 is as follows:

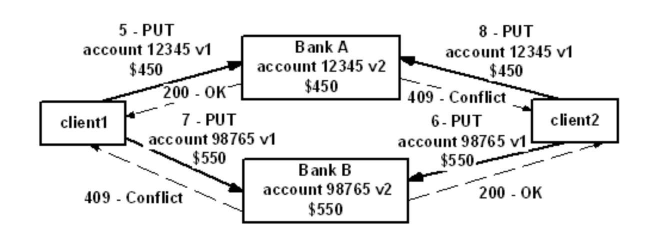


Figure 2: Each client can update only one of the manipulated resources and conflicts are detected.

```
PUT /accounts/account/12345/1 HTTP/1.1
Content-Type: application/xml
Host: bank.com
Content-Length: 138
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<account>
<accountNumber>12345</accountNumber>
<balance>450</balance>
<version>1</version>
</account>
```

It is important to note that the update URI is formed by the concatenation of the URI that identifies the account, `/accounts/account/12345`, with the version it want to change, `/1`.

With the detected conflicts, each client must make the compensation of changes already made. Thus, client 1 retrieves the just created version of the account 12345 and client 2 retrieves the just created version of the account 98765, to carry out the compensation stage. See Figure 3.

Now, with the new versions, the clients make changes locally and submit the compensation updates. See Figure 4. Therefore, the accounts 12345 and 98765 return to their original balance maintaining the integrity of the state.

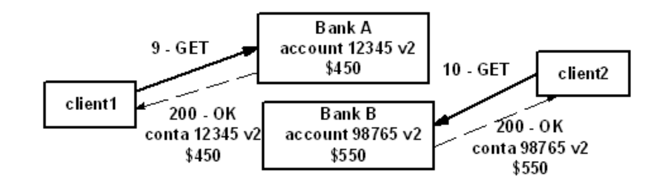


Figure 3: The clients identify the conflicts and retrieve the new versions to compensate them.

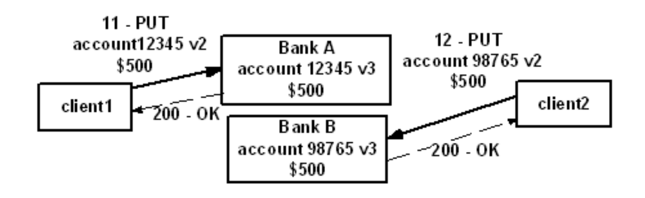


Figure 4: The clients compensate the earlier completed operations.

As it is seen, the proposal uses the concept of *compensation*, i.e., the act of undoing previously successfully completed work. For example, the value of $50 was credited to the account 12345 again by the client 1. Client 2 debited $50 of the account 98765 since he failed to complete the transaction.

It is interesting to established a contract of use between clients and RESTful servers in order to prevent that malicious or defective clients can make updates, even receiving a conflict notification triggered by the server. In this way, a service level agreements could specify the security policy that clients and serves must obey to interact. Thus servers are protected from malicious or defective clients and vice-versa.

2.4 Compensation

In general, as indicate by Gray [3], early exposure of uncommitted data is essential for long-duration and nested transactions. In [4], it is shown the method of transactions compensation for applications that need to both expose data not committed and undo transactions committed. At the end of a transaction compensation, it is guaranteed consistent state is established based on semantic information.

Compensation is one of the ways to design and implement models of transactions for web services. According to [12], unlike the traditional distributed transactions, the transactions between web services do not follow all the ACID properties. Due to weak coupling and autonomy required by web services, traditional protocols, for example, two phase commit protocol, are not appropriate.

Moreover, the transaction involving several web services can take considerable time to finish, minutes, hours or even days, which makes it impracticable to block resources used by web services, such as databases. When transactions have long duration, it makes no sense to prevent access to data that have not yet been committed, forcing other transactions to wait until the transaction that is running commits [4].

Therefore, it is possible that partial results of the transaction become available to other transactions. Those transactions that read and possibly alter the partial results of the overall transaction T are called *dependent transactions*. In this case, we say that T has been externalized. The purpose of compensation is to handle situations where we want to undo a externalized transaction T, without resorting to cascading aborts.

A formal way to examine a compensated execution is by comparing it to a hypothetical execution of only the dependent transaction, without the compensated transaction. A compensated execution that obeys this rule is one in which compensating operations commutes with operations of dependent transactions. Commutativity is a key notion in the context of compensation [4].

Thus, if some object is only manipulated with additions and subtractions, and if the log records the delta rather than the old and new value, then compensating operations may be commutable with dependent transactions operations [3]. It is the case of the example showed in Section 2.3.

The key point of using the compensation recovery paradigm is that we would like to leave the effects of the dependent transactions intact while preserving the overall state consistent, when undoing a transaction [4].

The presented procedure fits in REST style because all resources have a common interface. It is possible to suggest a mapping between the actions of possible compensation and the set of basic operations that can be used to handle a REST resource. In the paper proposal, considering the four operations CRUD, the possible compensations are:

- If the client creates a resource, the corresponding compensation operation is the act of erasing the created resource;
- If the client changes state X of a resource to state Y, the compensation operation is the act of undoing the change using, preferably ,the delta between X and Y;
- If the client deletes a resource, the corresponding compensation operation is the act of recreating the resource in the same state it was deleted.

The read operations require no compensation. The client has the responsibility for both control and manage the operations of compensation when something goes wrong. Every state of the application remains on the client, who is responsible for deciding to start the compensation when necessary.

Despite the mapping of compensating operations above, it is worth to note that compensation is an application-dependent activity [4]. For example, the application's semantic defines if the commutativity can be reached in order to apply the compensation recovery paradigm. Therefore, the suggested mapping has not the intention to serve for all applications types.

3. ANALYSIS OF THE PROPOSED TECHNIQUE

Through the implemented example, we verified that the ACID properties are observed when more than one transaction manipulates only one resource. Atomicity is achieved because only two situations can occur: client can change the resource or a conflict is detected.

The validation phase, described in Subsetion 2.2, responsible for verifying that the version being changed is the current version, ensures consistency, because only valid transitions between states are allowed.

As changes are first made in the resource local copies, isolation property is respected, because other clients do not have access to account 12345 changes while changes are not sent and validated in bank A's web service.

Assuming banks A and B store accounts information within persistent database, database management system, durability is achieved at the end of the transaction.

On the other hand, when considering the scenario where there are two clients concurrently manipulating two resources (or more), the following considerations can be made regarding ACID properties.

In the example, when the client 1 obtains copies of accounts, he locally does changes and tries to send the changes to the respective banks. As it was seen, it is possible to occur a case where account 12345 is changed in the bank A, but changes in account 98765 in bank B is not allowed, because of a conflict. Thus, the client 1 have to compensate the account 12345. However, before the client 1 can do it, another client may have retrieved the account information with the wrong balance and makes changes. In this case, the isolation is not reached, but it causes no major consequence, because at some point the compensation takes place and the balance of that account represents all the debits and credits actually made.

Atomicity is achieved through the use of compensation. If the update of one account fails, the succeeded updates have to be undone. Consistency property is obeyed, because when the transaction ends with success or making compensation, the final state of the accounts is valid.

In addition to examining the ACID properties, some considerations about the performance are raised. The performance is proportional to the number of detected conflicts. A major negative impact on the performance occurs in the worst case described for the optimistic concurrency control, where the number of conflicts is very high.

During the implementation to check proposal feasibility, we verified that the number of conflicts is greater when occur race conditions situations, which various clients attempt to

update two or more resources concurrently. In those cases, an update made by one client repeatedly causes transaction compensation in another client and vice-versa. Thus, none of them can finish the complete transaction. To avoid the race, some recommendations are described below.

In the example implementation, was assumed no pre-established order for services invocation, and thus, it is interesting to assume default invocation order for resource updates (HTTP PUT). The purpose of this recommendation is to avoid race conditions, in which there is a long dispute for resource updates concurrently.

For transactions involving various web services invocation without a pre-established order, a good practice is: (1) The transaction gets all URIs of RESTful Web services necessary for its execution. It is necessary a prior knowledge of all services that are used; (2) Order the URIs alphabetically. Since URIs are just text all transactions using the same web services obtain the same list of ordered URIs; (3) Process the transaction execution invoking services in the obtained order.

Therefore, in the example, both clients try to make updates in the same order. The conflict is detected in the first involved account and the update request of the second account is not sent. Consequently, the compensation is avoided.

The use of a random wait time can also be used to reduce the number of conflicts. Clients should wait a certain time period between fail update attempts.

4. CONCLUSIONS

Transactions implementation for RESTful web services using the optimistic concurrency control and the transaction compensation methods is relatively simple and obeys the REST architectural style. The architectural style imposes some restrictions for implementation of web services, complementing the service oriented architecture (SOA).

The example of transferring funds between bank accounts allowed to demonstrate the proposal feasibility. For the proposal, ROA is used, which is a concrete architecture to implement web services that conform to REST.

The usage of optimistic concurrency control method provides some advantages, for example, it is deadlock-free, as it does not use locks to information access control. But there are some disadvantages, such as the possibility of starvation. For example, one client may always get resource updates before other clients, who never finishes a resource update successfully. One way to alleviate the problem, is that after each satisfactory PUT request, the client waits for a random time before submitting a new PUT request to the same resource. The goal is to allow other clients to accomplish their resource updates.

For future work, we intend to apply the concurrency control method based on timestamp to monitor transactions between RESTful services and carry out the comparison with the results achieved in this paper.

We also intend to address issues of infrastructure fault tolerance such as host failure, broken link, web server and database unavailability. For client failure, active transactions may be salvaged across client-systems restarts by using transaction save points [3]. A transaction declares a save point and the transaction is reset to its most recent save point in the event of a system crash (restart).

For server failure, considering interactions are stateless, simplified techniques could be used. For example, a basic mechanism that detects the fault and directs future requests to redundant servers. Despite these mechanisms are not capable of tolerating faults while processing a request, for the most RESTful web services it should be sufficient. For a more robust solution, one could use the proposal of FTWeb [11]. The fundamental idea for the FTWeb model is deploying the active replication technique to achieve fault tolerance in service oriented architecture.

5. REFERENCES

[1] R. L. Costello. Building web services the rest way, s.d. `http://www.xfront.com/REST-Web-Services.html` access date March 2008.

[2] R. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, USA, 2000.

[3] J. Gray. The transaction concept: Virtues and limitations. In *Proc. Int'l. Conf. on Very Large Data Bases*, page 144, Cannes, France, Sept. 1981.

[4] H. F. Korth, E. Levy, and A. Silberschatz. A formal approach to recovery by compensating transactions. In *VLDB '90: Proceedings of the 16th International Conference on Very Large Data Bases*, pages 95–106, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc.

[5] H.-T. Kung and J. T. Robinson. On optimistic methods for concurrency control. *ACM TODS*, 6(2):213–226, June 1981.

[6] OASIS. Oasis web services reliable messaging (wsrm) tc, November 2004. `http://www.oasis-open.org/committees/tc\_home.php?wg\_abbrev=wsrm`.

[7] OASIS. Oasis web services security (wss) tc, February 2006. `http://www.oasis-open.org/committees/tc\_home.php?wg\_abbrev=wss`.

[8] OASIS. Oasis web services transaction (ws-tx) tc, July 2007. `http://www.oasis-open.org/committees/tc\_home.php?wg\_abbrev=ws-tx`.

[9] OASIS. Web services coordination (ws-coordination), July 2007. `http://docs.oasis-open.org/ws-tx/wscoor/2006/06`.

[10] L. Richardson and S. Ruby. *RESTful Web Services*. O'Reilly & Associates, Sebastopol, California, May 2007.

[11] G. T. Santos, L. C. Lung, and C. Montez. FTWeb: A fault tolerant infrastructure for web services. In *EDOC*, pages 95–105. IEEE Computer Society, 2005.

[12] T. Strandenæs and R. Karlsen. Transaction compensation in web services. *Norsk Informatikkonferanse - NIK*, november 2002.

[13] W3C. Naming and addressing: Uris, urls, ... `http://www.w3.org/Addressing/URL/uri-spec.html` access date March 2008.

[14] W3C. Simple object access protocol (soap) 1.1, May 2000. `http://www.w3.org/TR/2000/NOTE-SOAP-20000508/`.

[15] W3C. Web services description language (wsdl) 1.1. Note, March 2001. `http://www.w3.org/TR/2001/NOTE-wsdl-20010315`.

EistCockpit

Anhang A

Sitzungsprotokolle, Risikomanagement

David Schöttl, Remo Waltenspül & Diego Steiner

A.1 Sitzungprotokoll 20.02.2012

A.1.1 Traktanden

Anwesend: mstolze, tcorbat, dschoett, rwaltens, dsteiner

Abwesend: -

- Aufabenstellung
- Zusammenarbeit klären

A.1.2 Aufgaben

- WK Einsatz für rwaltens (tcorbat)
- Aufgabenstellung (mstolze)
- Infrastruktur aufsetzen (dsteiner)
- Richtlinien definieren (Projektteam)
- Projektplan grob definieren (Projektteam)

Nächstes Meeting: Mittwoch 22.03.2012

A.1.3 Rückblick

-

A.1.4 Protokoll

A.1.4.1 Name

Der Name “EistTm Cockpit” wurde von allen Beteiligten akzeptiert.

A.1.4.2 Übereinkünfte

Die Ausführenden halten in einem Meeting die Richtlinien für Sprache (Dokumentation), Terminologie, Tools, usw. und erstellen ein entsprechendes Dokument.

A.1.4.3 Aufgabenstellung

MS leitet aus dem Email von dschoet die vollständige Aufgabenstellung ab und signiert diese, dass man sie als Grundlage für die SA nehmen kann. Deadline ist der 26.02.2012.

A.1.4.4 Zusammenarbeit

rwaltens (z.Z. Aufklärer bei InfBat 6) ist nicht für den WK eingeteilt. tcorbat versucht noch kurzfristig eine Teilnahme am WK bei Hptm Klötzli zu erwirken.

Die Hoffnung besteht, dass wir (wenn auch einen trivialen) Internetzugang etablieren können, um die Kommunikation auf elektronischem Weg sicherzustellen, falls die Umteilung von rwaltens nicht klappt.

tcorbat klärt ebenfalls ab, ob die Beteiligten Personen eine Geheimhaltungserklärung unterzeichnen müssen.

A.1.4.5 Infrastruktur

Die HSR würde einen Server (Debian, VM) für die Arbeit zur Verfügung stellen. DST setzt die Infrastruktur bis Ende Woche auf.

A.1.4.6 Umfang und Erwartungen

Ziel wäre es, dass der Umfang des Projektes für eine BA reicht. Falls sich abzeichnet, dass der Umfang zu klein werden sollte, wird der Umfang durch zusätzliche Features angepasst.

A.1.4.7 Meetings

Die Regel soll ein Progress-Meeting pro Woche stattfinden. Als Wochentag wurde der Mittwoch um 15:00 Uhr bestimmt. Andere Meetings werden je nach Bedarf jederzeit durchgeführt.

Aufgrund des WKs wird das nächste Meeting am Freitag, 24.02.2012 um 15:10 in der IFS stattfinden.

A.1.4.8 Rechte am Code

Die Regelung der HSR sollte für unseren Gebrauch gut geeignet sein. Allenfalls muss das Projekt als Opensource lizenziert werden, um die Transparenz zu gewährleisten. Andereseits unterliegt der Code vielleicht einer gewissen Geheimhaltung, wobei dann eine geschlossene Lizenz zu gebrauchen wäre. TC klärt noch ab, was die Interne HSR Regelung angeht.

A.1.4.9 Bewertung

Die Bewertung erfolgt mittels der Bewertungsmatrix von MS. Über Formulierung und Gewichtung kann bis zur Ende der Arbeit diskutiert werden.

Von Vorteil wäre eine kurze Video-Präsentation der Arbeit, wobei aber noch abgeklärt werden muss, ob das in unserer Umgebung mit Geheimhaltung möglich ist.

A.1.4.10 Abgabe

Ein grosses PDF (bzw. 2: Technisch/Management) für Alles, durchnummeriert bis zum Anhang.

A.1.4.11 Verbindung mit UInt2

Die Verbindung mit UInt2 würde sich anbieten und ist von MS bewilligt. Die wirkt sich aber nicht auf die SA-Note aus.

A.1.4.12 Tipps

- Fokus auf Code, Testing & Documentation statt auf zBsp. perfekte Anforderungsanalysen.
- Kellersche Tipps beachten.
- Zeit gut einteilen, nicht herausschieben, Blockzeiten definieren

- Documentation-Driven Refactoring
- Kriterienliste genau beachten, lieber öfters Nachfragen
- Unterstützer sind da um bei Entscheidungen zwischen Alternativen zu helfen, nicht um Lösungsvorschläge zu generieren.
- Traktandenliste und Probleme für das folgende Meeting vorbereiten.

A.2 Sitzungsprotokoll 22.02.2012

A.2.1 Traktanden

Anwesend: mstolze, tcorbat, dschoett, rwaltens, dsteiner

Abwesend: -

- Übereinkünfte treffen
- Zeitplan erstellen

A.2.2 Rückblick

- ~~WK Einsatz rwaltens (tcorbat)~~ Erledigt: rwaltens kann teilnehmen
- Aufgabenstellung (mstolze)
- ~~Infrastruktur-Server aufsetzen (dsteiner)~~ Erledigt: Infomail folgt
- Richtlinien definieren (alle): Subjekt dieses Meetings
- Projektplan grob definieren (alle): Subjekt dieses Meetings

A.2.3 Aufgaben

- MS und Sprints in Redmine erfassen (dsteiner)
- Loginmails versenden
- Kleine Architekturübersicht erstellen (dschoett): Siehe Architecture.
- Updates mstolze via tcorbat und Redmine

A.2.4 Protokoll

A.2.4.1 Übereinkünfte

- Namenskürzel ist HSR Login; konkret:
  - dschoett (Dave Schöttl)
  - dsteiner (Diego Steiner)
  - rwaltens (Remo Waltenspül)
  - mstolze (Markus Stolze)
  - tcorbat (Thomas Corbat)
- Dokumentationssprache Deutsch, Wiki ist gemischt deutsch/english.
- Codedoku mit Doxygen
- Als SCM wird SVN eingesetzt
- Als Kollaborationtag sei der Montag Nachmittag ab 1300 Uhr festgesetzt

A.2.4.2 Vorgehensmodell

- Design und Planungsphase: RUP Orientiert
- Entwicklung des Prototyps: SCRUM Orientiert

A.2.4.3 Benutzerbeobachtung

- Ist-Analyse erstellen
- Szenarien und Personas erstellen

A.2.4.4 Zeitplanung

Milestones

RUP Ortientiert

- MS1 Affinity & Personas
- MS2 Anfordungen und Analyse
- MS3 Personas & Szenarien

SCRUM Orientiert

- MS4 GUI Prototyp und Testresults
- MS5 Architekturprototyp
- MS6 End of Elaboration
- MS7 Architektur & Design
- MS8 Beta Release
- MS9 Final Release
- MS10 Dokumentation

Der Plan ist dass wir im WK die gesamte Anforderungsanalyse (MS1 bis MS3) fertigstellen. Ab dann wollen wir auf ein agiles Vorgehen umsteigen. Bis Ende Woche 17 ist die Architektur und das Design spezifiziert. Bis Ende Woche 22 ist die Software in einer ersten finalen Version fertiggestellt und dokumentiert. Der Projektplan wird noch genauer durch dsteiner spezifiziert.

A.2.4.5 Wichtige Punkte aus der "Matrix of Doom"

Technologie Entscheid ab MS4

A.3 Sitzungprotokoll 24.02.2012

A.3.1 Traktanden

Anwesend: dschoett, rwaltens, tcorbat, mstolze, dsteiner

Abwesend: -

- Grobplanung besprechen
- Vorgehensmodell besrechen
- MoD Punkte besprechen
- Grobarchitektur besprechen, siehe: Architecture
- Update von mstolze regeln

A.3.2 Aufgaben

- Sprints auf 3 Wochen verlängern (dsteiner)
- "IT-Sackbefehl" vorbereiten

A.3.3 Rückblick

- Protokoll vom 20.02 ist von allen Parteien abgenommen worden
- Protokoll vom 22.02 ist von allen Parteien abgenommen worden

A.3.4 Protokoll

A.3.4.1 Updates von mstolze

Hauptbetreuer ist sowieso tcorbat, also läuft die Kommunikation über Ihn

A.3.4.2 Vorgehensmodell

Unser Hybridmodell ist soweit in Ordnung, vorallem da der Kontakt mit dem Kunden vor allem in den WK Wochen vorhanden sein wird. Die Sprints sind jedoch für unser Zeitbudget viel zu kurz, diese müssen auf 3 Wochen/Sprint verlängert werden.

A.3.4.3 Geheimhaltung

Der Auftraggeber verlangt, dass Schützenswerte Inhalte aus der Arbeit herausgehalten werden. "Alles was nicht sowieso über die Webseite verfügbar ist und nicht als nicht Schützenswert gilt kann verwendet werden". Dazu gilt noch ein Dokument zu erstellen, dass die TEPLAS Umgebung den Anforderungen entsprechend beschreibt. Das Zugehörige Dokument wird von tcorbat erstellt.

A.3.4.4 Architektur

Allgemein

Das TEPLAS ist auf Microsoft Technologie aufgebaut, es gilt also die Anforderung die Applikation auf diese Umgebung anzupassen. Konkret bedeuted das den

Einsatz von .net.
Die Grundsätzliche Grobarchitektur mit den Tiers [Server, Triage-Messaging-Client und Cockpit-Fat-Client] ist soweit in Ordnung, wobei wir uns im Rahmen der SA vA. auf den Server-teil und Triage-Client konzentrieren werden müssen

Userverwaltung

Eine Benutzerverwaltung wird ausgeklammert, da diese sowieso implizit über das Active Directory und den Windows-Login abgefangen wird.

A.3.4.5 Bewertungsmartrix

- Punkt 1.2, 1.16: Es wird kein Video erstellt, ein externer Wiki Eintrag dürfte nach Absprache möglich sein
- Punkt 2.14: Ziel ist es, richtig zitieren zu können, wie auch die HSR Libraries zu diesem Thema zu kennen. Der Punkt bleibt drin
- Punkt 3.1.3.8: Der Punkt ist zumindest zu diskutieren (und Ergebnis festhalten), muss aber nicht "implementiert" werden
- Punkt 3.3.2.4: Doxygen erfüllt die Auflagen

A.3.4.6 WK Vorbereitungen

Anwesenheit von tcorbat während des WKs

- KVK - W1: Mo bis Mi
- W2 - W3: Di bis Di

A.4 Sitzungprotokoll 05.03.2012 (14:00)

A.4.1 Traktanden

Anwesend: dschoett, rwaltens, tcorbat, dsteiner

Abwesend: -

- Namensänderung "EistCockpit"
- Arbeiten für nächsten 3 Woche
- Matrix of Doom nachbesprechen (Wie Punkte gewährleisten)
- Geheimhaltungs-Erklärung (Corbat)
- Accessibility
- Review Aufgabenstellung
- Präsentationspunkte besprechen

A.4.2 Aufgaben

Nächstes Meeting: Kick-Off Meeting 05.03.2012 19:00

- Präsentationsfolien für Meeting erarbeiten
- Alle Projektmitglieder reviewen Aufgabenstellung

A.4.3 Rückblick

- Protokoll vom 24.02.2012 muss noch abgenommen werden!

A.4.4 Protokoll

A.4.4.1 Namensänderung "EistCockpit"

Da bereits eine Anwendung mit der Bezeichnung 'TmCockpit' existiert, wurde im kollektiv
bestimmt, dass die zu entwickelnde Applikation den Namen 'EistCockpit' tragen soll.

A.4.4.2 Arbeiten für nächsten 3 Wochen

1. Woche

- IST-Situation aufnehmen (3 Personas Triage, 2 Personas TBZ/Eist)
- Benutzerbefragung (User-Stories mit Interviews)
- Review Projektplanung/Matrix of Doom (MoD)

2. Woche

- Visionsdokument erstellen
- Soll-Situation definieren
- Paper-Prototyping
- Priorisierung Module

3. Woche

- Einkalkulierte Pufferzeit

A.4.4.3 Matrix of Doom nachbesprechen

Der erste Abschnitt wurde im Team besprochen. Ziel war es das Vorgehen um die Bewertungskriterien zu gewährleisten klar zu definieren. Die noch nicht im Plenum diskutierten Punkte werden in den kommenden Tagen weiter bearbeitet.

A.4.4.4 Geheimhaltungs-Erklärung (Corbat)

Thomas Corbat hat uns eine erste Version der Geheimhaltungs-Erklärung unterbreitet. Zudem erläuterte er kurz die wichtigsten Punkte:

- Schützenswerte Informationen dürfen unberechtigten nicht zugänglich gemacht werden.
- Reale Daten mit entsprechenden Klassifikationen dürfen nicht in die Datenbank abgelegt werden.
- Projekt-Dokumentation darf keine Informationen der genannten Klassifizierung enthalten.
- Im Zweifelsfall entscheidet Bataillonskommandant Major Hans-Andrea Veraguth
- Projektbeteiligten verpflichten sich, Kentnisse über Personen oder das System vertraulich zu behandeln.

Dieses Dokument wird von sämtlichen Projektmitgliedern, sowie dem Projektauftraggeber unterzeichnet.

A.4.4.5 Accessibility

Folgende Fragen wurden für das kommende Kickoff-Meeting aufgelistet:

- Was sind mögliche Einschränkungen bzw. Handicaps von Nutzern (AdA), welche bei der Entwicklung beachtet werden müssen?
- Sind Personen mit Farbblindheit oder Rot/Grün Schwäche trotzdem Diensttauglich?

A.4.4.6 Review Aufgabenstellung

Die Aufgabenstellung wurde von rwaltens, dschoett und dsteiner akzeptiert.

A.4.4.7 Präsentationspunkte vorbereiten

Vor der Erarbeitung der Präsentationsfolien wurde kurz der Inhalt bzw. der logische Aufbau definiert. Die Traktanden für das Kickoff-Meeting und folgedessen der Präsentation sind wie folgt:

1. Aufgabenstellung
2. Vorteile für das Ristl Bat
3. Architektur
4. Accessibility
5. Nachfolgeregelung & Maintenance

6. Projektplanung während WK
7. Konsequenzen
8. Geheimhaltungs-Erklärung

A.5 Sitzungprotokoll 05.03.2012 (19:00)

A.5.1 Traktanden

Anwesend: dschoett, rwaltens, tcorbat, dsteiner, Hptm P. Furrer, Maj H.-A. Veraguth, Oblt B. Helfenberger

Abwesend: -

- Aufgabenstellung
- Vorteile für das Ristl Bat
- Architektur
- Accessibility
- Nachfolgeregelung & Maintenance
- Projektplan während WK
  - Woche 1
  - Woche 2
  - Woche 3
- Konsequenz
- Geheimhaltungs-Erklärung

- Fragen

A.5.2 Aufgaben

Nächstes Meeting: Montag 12.03.2012

- Q: Wie wird sichergestellt das auch in Zukunft der Code für Erweiterungen etc. zur Verfügung steht?
  - A: Vorschläge werden von dsteiner unterbreitet
- Q: Wer ist für die künftige Weiterentwicklung des Systems verantwortlich? Ristl Bat? FU Br?
  - A: Momentan noch nicht von gravierender Wichtigkeit. Fw Schöttl wird noch mehrere Jahre im Ristl Bat 20 dienen, daher wird vorerst die Weiterentwicklung durch ihn im Rahmen der KVKs/WKs erfolgen (im Sinne eines spez Det).
- Q: Wie verhält sich System wenn Notizen nicht digital erfasst werden?
  - A: Frage muss während der Analyse der IST-Situation anhand von Interviews & Personas geklärt werden. (rwaltens & dsteiner)
- Q: Wie reagiert System bei Absturz des TEPLAS-Server?
  - A: Es muss abgeklärt werden, wie hoch die Verfügbarkeit des TEPLAS-Server ist, sowie ein Vorgehen im Falle eines Absturzes definiert werden.

- In den nächsten Tagen Interviews mit mehreren potentiellen Benutzern führen.

A.5.3 Rückblick

- Protokoll vom 05.03.12 14:00 Uhr muss noch abgenommen werden.

A.5.4 Protokoll

A.5.4.1 Fragen zu den Anforderungen

Besonders in hektischen Situationen werden oftmals Notizen nicht über den digitalen
Umweg bearbeitet. Wie verhält sich das System unter den gegebenen Umständen?

- Es muss sichergestellt werden das der Fall abgedeckt wird. Der Umfang wird im Rahmen von Interviews und Personas geklärt.
- Wie reagiert das System im Falle eines Absturzes des TEPLAS-Server?
- Es gilt abzuklären wie hoch die Anforderungen bezüglich der Verfügbarkeit des Servers sind. Zudem soll definiert werden, wie sich das System bei Ausfällen verhält.

A.5.4.2 Fragen zur Installation/Infrastruktur

- Die Details zur Installation der Anwendung auf dem TEPLAS Server muss nochmals mit der LBA genauer geklärt werden.
- Bestehen Richtlinien bezüglich eingesetzter Technologien aufgrund von vorgesehenen Konsolidierungen von Militärsoftware?
  - Dieser Punkt muss bei der schweizerischen Armee noch abgeklärt werden.

A.5.4.3 Nachfolgeregelung & Maintenance

- Wer ist der Ansprechpartner nach dem Abtritt von Maj Veraguth?
  - Gem. Aussage von Maj Veraguth wird diese Aufgabe dem neuen Bat Kdt Stv (Hptm R. Berger) übertragen.
  - Gleiches gilt für den S3: Der aktuelle S3 Hptm Schneiter ersetzt den ehemaligen S3 Hptm Klötzli.

A.5.4.4 Analyse IST-Situation

- Aktuelle Formulare (Meldezettel etc.) können eingescannt werden für die Dokumentation.
  - Wichtige Anmerkung: Es dürfen keine Meldezettel mit einer Klassifizierung (intern, geheim, vertraulich) benutzt werden.
- Möglichst viele verschiedene zukünftige Benutzer interviewen, um ein umfassendes Bild der gegenwärtigen Situation zu erhalten.

A.5.4.5 Architektur

- Wieso wurde der Entscheid zugunsten einer Client/Server und nicht einer Webbasierten-Architektur gefällt?

  - Aufgrund der aktuell existierenden System-Umgebung, dem Wissensstand der Projektmitarbeiter und - da keine Vorgaben diesbezüglich gemacht wurden - die Präferenz der Projektmitarbeiter. Ziel ist es, eine brauchbare Software zu entwickeln, und nicht, neue bzw. mühsame Technologie (Erfahrungs-Wert) einzusetzen.

A.5.4.6 Accessibility

- Farbenblindheit sollte kein Problem darstellen, da in der schweizerischen Armee verhältnismässig etwa gleichviele Farbenblinde wie in der gesamten schweizer Bevölkerung existieren. Deshalb sollte nicht nur auf farbliche Indikatoren gesetzt werden um bestimmte Programmoptionen zu unterscheiden; z.B. durch zusätzliche text-basierte Anzeigen (grün + "OK", rot + "FEHLER").

A.5.4.7 Geheimhaltung

Laut Maj Veraguth ist es äusserst wichtig, dass keine Daten mit einer militärischen Klassifizierung in den Projektdokumentationen
vorkommen. Dabei gelten insbesondere die folgenden Regeln, welche aus der Geheimhaltungsvereinbarung stammmen:

- Handhabung von Informationen der genannten Klassifizierungen müssen berücksichtigt werden.
- Reale Daten mit entsprechenden Klassifikationen dürfen nicht in Datenbank abgelegt werden (fiktive Daten verwenden).
- Projekt-Dokumentation enthält keine Informationen mit offiziellen militärischen Klassifizierungen (intern, vertraulich, geheim).
- Im Zweifelsfall entscheidet Stellvertretender Bataillonskommandant Maj Veraguth
- Projektbeteiligten verpflichten sich, erlangte Kenntnisse über Personen oder System vertraulich zu behandeln.

A.5.4.8 Arbeit während dem WK

- Es wurde durch Maj Veraguth versichert, dass im Minimum ein Zeitkontingent von 4-5 Arbeitsstunden/Woche während dem WK ermöglicht wird.

  - Anmerkung: Der militärische Auftrag muss selbstverständlich trotzdem einwandfrei erfüllt werden.

A.5.4.9 Feedback zur Präsentation

- Einzelne Passagen während der Präsentation waren gespickt mit Füllwörtern ("äh, hm etc."); diese sollten zugute des Präsentationsflusses vermieden werden. (Hptm Furrer)

- Inhalt der Präsentation sollte kurz erläutert werden, damit klar ist, was besprochen wird. (Oblt Helfenberger)
- Bei der Nutzung des Laserpointer, sollte man darauf achten, dass man keinen nervösen Eindruck macht, indem man schnell auf verschiedene Dinge zeigt. (Oblt Helfenberger)
- Eventuell sollten englische Fachausdrücke besser erläutert werden, damit auch Personen mit weniger hohem technischem Verständnis den Inhalt verstehen. (Hptm Furrer)

A.6 Sitzungsprotokoll 26.03.2012

A.6.1 Traktanden

Anwesend: dschoet, rwaltens, dsteiner

Abwesend: -

A.6.2 Rückblick WK

- Projekts- und Qualitätsmanagement
- Redmine Cleanup
- Stunden buchen

A.6.3 Planung

- UI Prototyp besprechen

A.6.4 Aufgaben

- Upcoming Deadlines: Abnahme Phasen: 28.03.2012

A.6.5 Protokoll

- Arbeit gemäss Traktandenliste

A.7 Sitzungsprotokoll 28.03.2012

A.7.1 Traktanden

Anwesend: dsteiner, rwaltens, tcorbat

Abwesend: dschöttl (Krank), mstolze

- Abnahme der Sitzungsprotokolle und der Phasen
- Rückblick WK
    - gesteckte Ziele
    - erreichte Ziele
    - Auswertung
    - Risikomanagement
    - Konsequenzen
- Ausblick, Planung

A.7.2 Aufgaben

- Alle Dokumente überarbeiten (reviewen)
- Abgabedatum bei allen Dokumenten anpassen
- Abklären wie hoch Verfügbarkeit von Teplas Server ist
- Backlog erstellen
- Überlegen wie auf einen Stromausfall reagiert wird.

A.7.3 Protokoll

A.7.3.1 Abnahme der Sitzungsprotokolle und der Phasen

- Sitzungsprotokoll 22.02.2012 wurde von Thomas Corbat abgenommen.
- Sitzungsprotokoll 24.02.2012 wurde von Thomas Corbat mit folgenden Einwänden abgenommen.
    - Geheimhaltungsdokument wird von Thomas Corbat erstellt (explizit erwähnen)
    - Zitate aus der HSR Libraries (IEEE oder wissenschaftlichen Berichts) können nicht durch Militäre Dokumente ersetzt werden
    - Fallback unbedingt implementieren, im Falle eines Stromausfalles.
    - Ev. Stündlich Meldungen ausdrucken und zuständigen Personen in Fächer legen
    - Termine der Anwesenheit von Thomas Corbat nicht ganz korrekt.
    - Kapitel Architektur beschreibt ein System mit einem Webinterface, dies wurde jedoch durch ein Fat-System ersetzt.
- Sitzungsprotokoll 05.03.2012 (14:00 Uhr) wurde von Thomas Corbat abgenommen.
    - Präziser spezifizieren wer alles die Aufgabenstellung reviewed.

- Sitzungsprotokoll 05.03.2012 (19:00 Uhr) mit folgenden Einwänden abgenommen.
    - Verfügbarkeit von Teplas-System muss noch abgeklärt werden
    - 3. Frage noch nicht bekannt, wie Meldungen auf nicht digitalem Weg erfasst werden
    - Einzelner Punkt anpassen, wer ist Entscheidungsorgan Funktion & Person (sowie nachfolgende zuständige Person)

A.7.3.2 Abnahme der Projektphasen

- Inception wurde durch Thomas abgenommen.
- Weitere Phasen enthalten noch Arbeitspakete (vorwiegend Reviews) und werden deshalb beim nächsten Meeting abgenommen.

A.7.3.3 Rückblick WK

- Diego Steiner & Remo Waltenspül berichten kurz über die Eindrücke und positiven Erkenntnisse während dem WK.

A.7.3.4 Risikomanagement

- Risikomanagement Anmerkungen von Thomas
- Weitere Risiken Produktspezifisch (Bsp. Produkt wird nach Fertigstellung abgelehnt)

A.7.3.5 Papierprototyp

- Papierprototyp wird an Thomas auf Benutzbarkeit erprobt

A.7.3.6 Zeitplanung

- Antrag zur Verlängerung der Arbeitszeit für Studienarbeit um 1 Woche genehmigt.
    - Impliziert neuen Abgabgetermin: Freitag 08.06.2012
- Frage von Thomas, ob geplante Arbeitszeit wirklich realisierbar ist
    - Diego Steiner & Remo Waltenspül bestätigen, dass Zeit für die Durchführung der Arbeit reichen sollte.
    - Anmerkung von Thomas: Diagramm mit akkumulierter Zeit für übersichtlichere Darstellung

A.7.3.7 Allgemeine Anmerkungen

- Namen der Interviewpartner anonymisieren
- Diverse Rechtschreibefehler in Dokumenten
- Gültigkeitsbereich in Visionsdokument anpassen
- Thomas soweit zufrieden mit bisherigem Projektstand

A.8 Sitzungsprotokoll 04.04.2012

A.8.1 Traktanden

Anwesend: rwaltens, dsteiner, dschöttl, tcorbat

Abwesend: mstolze

- Abnahme des letzten Sitzungsprotokolls
- Abschluss SP0
  - Backlog
- Stand der Arbeiten SP1
  - Interpretation Session (Auffälligkeiten am Arbeitsplatz)
  - Affinity
  - Entwurf domain model
- Ausblick

A.8.2 Aufgaben

- Abklären wie hoch Verfügbarkeit von Teplas Server ist (dsteiner)
- Dokument layer architecture überarbeiten

A.8.3 Protokoll

A.8.3.1 Besprechung Schichtenarchitektur

- WCF auf gleicher Ebene wie Viewmodel (theoretisch keine eigene Schicht)
- Ansonsten keine Einwände bezüglich Schichtenarchitektur
- Hinweis von Thomas: Abkürzungen bei der ersten Verwendung ausschreiben, bei weiterem Gebrauch reicht Abkürzung (Eintrag in Glossar)

A.8.3.2 Abnahme letzten Sitzungsprotokoll

- Das Sitzungsprotokoll vom 04.04.2012 wurde von Thomas Corbat abgenommen.

A.8.3.3 Abschluss SP0

Backlog

- Dave erklärt kurz alle definierten Aufgaben im Backlog
- Aufgaben werden weiter in Subtasks aufgeteilt, welche als User Stories beschrieben werden.
- Thomas ist mit dem Backlog soweit zufrieden

A.8.3.4 Stand der Arbeiten SP1

- UInt2 Artefakten (Interpretation Session, Affinity) demonstriert
- Domain-Model kurz vorgestellt

A.8.3.5 Ausblick

Folgende Arbeiten sind während dem Sprint 1 vorgesehen

- Subtasks aus den im Backlog aufgelisteten Arbeitspaketen erstellen
- Domainmodel im Team erarbeiten
- Arbeitspakete DAL & Host Application implementieren
- Paper-Prototype entwerfen und mit Thomas testen

A.9 Sitzungsprotokoll 25.04.2012

A.9.1 Traktanden

Anwesend: dschöttl, tcorbat, rwaltenspuel, mstolze (teilweise)

Abwesend: dsteiner

- Abnahmen
  - Abnahme des letzten Sitzungsprotokolls
  - Abnahme SP1
- Prototypen zeigen (PoC)
  - UI Plugin Prototyp
  - WCF Prototyp
- Ausblick

A.9.2 Aufgaben

- Burndown-Chart erstellen
- Besprechen ob DoD (Definition of Done) sinnvoll ist und in Projekt eingesetzt werden soll.
- Schichtenarchitektur überarbeiten
- Architekturübersicht klarer illustrieren (Unidirektionale Verbindung)
- Prüfen, ob Vertragsnamen beim Pluginimport nötig sind
- Dokumentieren wie Plugins eingesetzt werden (Prozess, Kompetenzen)
- Überlegen, wo man die Hilfe anzeigen könnte

A.9.3 Protokoll

A.9.3.1 Abnahme der Sitzungsprotokolle

Sitzungsprotokoll 04.04.2012 wurde von Thomas Corbat mit folgendem Einwand abgenommen.

- Im Sitzungsprotokoll muss erwähnt werden, dass die Architektur auf einem Pluginsystem aufbaut und dadurch einfach erweitert werden kann.

A.9.3.2 Abnahme der Projektphasen

Sprint 1 wurde von Thomas Corbat mit folgender Anmerkung abgenommen.

- Burndown-Chart sollte unbedingt vorhanden sein, da es ein Kernartefakt von Scrum ist.

A.9.3.3 UI Plugin Prototyp

- Abhängigkeiten zu Host Application und weiteren Projekten verringern

    - Viewmodel besser in gleiches Projekt, dadurch werden Plugins unabhängiger

- Frage von Thomas, ob Vertragsnamen bei Pluginimport wirklich benötigt werden

    - Dave schlägt vor die Strings in einer XML Datei zu definieren, dadurch könnte man auf diese Referenz verzichten.

A.9.3.4 WCF Prototyp

- Thomas findet, es sei zu wenig klar aus dem Architekturdokument, dass man einen Webservice verwenden möchte.
- David berichtet über die Nachteile einer Lösung mit einem Webservice

    - Grundsätzlich handelt es sich eher um eine unidirektionale Verbindung
    - Um von Server Daten an Client zu übermitteln wird regelmässig Zustand abgefragt (Polling)
    - Vorschlag von David ca. jede Minute und vor kritischen Operationen (z.B. Bearbeiten) aktuelle Daten von Server an Clients schicken.

A.9.3.5 Ausblick

- Alle anstehenden Arbeitspakete im Sprint 2 werden Thomas kurz erläutert
- Ziel ist es am Ende des Sprint 2 eine erste Beta-Version fertiggestellt zu haben
- Remo fragt nach, ob es möglich ist, ein Codereview gegen Ende des 2. Sprints durchzuführen. Dies wird durch Thomas bestätigt.

A.9.3.6 Kurze Wiederholung Demo Prototypen

Anmerkungen durch Herr Stolze:

- Genau definieren wer die Plugins erhält und wie entschieden wird, welche wirklich benötigt werden.
- Die Hilfe Schaltfläche ist eventuell in der Pluginleiste schlecht positioniert
- Wie wird definiert, wer welche Rechte hat ohne im vornherein zu wissen, wie das Plugin aufgebaut ist. David beantwortet diese Frage umgehend, die Rollen des Active Directory werden auf die Applikationsrollen abgebildet und nur Standardoperationen (CRUD) angewendet.

A.10 Sitzungsprotokoll 02.05.2012

A.10.1 Traktanden

Anwesend: dsteiner, tcorbat, rwaltens

Die Sitzung wurde im Einvernehmen aller Beteiligten auf die folgende Woche verschoben.

A.11 Sitzungsprotokoll 09.05.2012

A.11.1 Traktanden

Anwesend: dsteiner, dschöttl, tcorbat, rwaltens

Abwesend: --

- Abnahmen
  - Abnahme des letzten Sitzungsprotokolls
- Stand der Arbeiten
  - Demo
  - Tasks für diese Woche
  - Zuteilung der Tasks
- Ausblick

A.11.2 Aufgaben

-

A.11.3 Protokoll

A.11.3.1 Abnahme der Sitzungsprotokolle

- Burndown chart: Stunden statt Storypoints nicht normal, aber ok
- Angepasstes architekturmodell
  - Read ist von Client initiierte Abfrage, ansonsten ok.
- Angepasste Layerarchitektur
  - Webserver ist immernoch WCF, nur zur clarification
- Begründungen und Entscheide gut dokumentieren

A.11.3.2 Tasks für diese Woche

- Stand der Arbeiten aufgezeigt

A.11.3.3 Ausblick

- Geheimhaltungserklärung nd aufgabenstellung für mstolze
- Abtretungserklärung ausfüllen
  - Kontakt FuBr heraussuchen

A.12 Sitzungsprotokoll 22.05.2012

A.12.1 Traktanden

Anwesend: dsteiner, dschöttl, tcorbat, rwaltens

Abwesend: --

- Abnahmen
  - Abnahme des letzten Sitzungsprotokolls
  - Abnahme des Sprints SP2
- Stand der Arbeiten
  - Demo der Betaversion
  - Tasks für diese Woche
  - Zuteilung der Tasks
- Ausblick

A.12.2 Aufgaben

A.12.3 Protokoll

A.12.3.1 Probleme

- Polling ineffizient
  - Andere Ansprechmoeglichkeiten des Webservice, zum Beispiel mit Diffs oder Timestamps
  - Performance mal mit 1000 Datensaetzen testen
- Validation
  - Stage Disclosure oder andere Patterns
  - Validation erst wenn fokus verloren oder beim Speichern

Alle Entscheidungen dokumentieren und sagen wohin die Zeit gegangen ist.

- Wieso DTOs?
- Wieso Polling?

A.12.3.2 SP2

- Im Sprint 2 sind wir mehre Tage hinderher

A.12.3.3 Abnahme der Sitzungsprotokolle¶

- Letztes Sitzungsprotokoll abgenommen.

A.12.3.4 Tasks für diese Woche

- Fertigstellen der Arbeiten
- Codereview
- Grobe Dokumentation

A.12.3.5 Ausblick

-

A.13 Sitzungsprotokoll 06.06.2012

A.13.1 Traktanden

Anwesend: dsteiner, dschöttl, mstolze, rwaltens, mgfeller

Abwesend: tcorbat

- Code Review: Feedback von Thomas
- Dokumentation:
  - Benutzer- & Installations-Manual
  - Kurzreview der bestehenden Dokumente
- Abtretungs-Erklärung
- Aufgabenstellung

A.13.2 Aufgaben

A.13.3 Protokoll

MoD

- Arbeit jetzt konstant mit Matrix of Doom

Folgearbeit

- Von Seite HSR angenommen
- Von Seite Militaer ist ebenfalls Unterstuezung vorhanden

Demo der Applikation

- AD anbindung fehlt
- Adminrechte im Plugin definieren?
- Icons: Rechte zum benutzen vorhanden? -> abklaeren!
- Maximierung: Beschraenken?
- Aufoesung der Laptops beachten

Code Review

Siehe Code Review Dokument

Änderungsgeschichte

Datum	Version	Änderung	Autor
07.03.2012	1.0	Erste Version des Dokuments	WR
12.03.2012	1.1	Dokument um Spätesten Zeitpunkt aktualisiert	WR
26.03.2012	1.2	Risiko 03 eingetroffen, Massnahmen beschrieben	WR
27.03.2012	1.3	Kleine Anpassungen bei Berechnung der Reserve	WR
22.04.2012	1.4	Bereinigen von Risiken nach Sprint 1	WR
20.05.2012	1.5	Bereinigen von Risiken nach Sprint 2	WR
06.06.2012	1.6	Bereinigen von Risiken gegen Ende Sprint 3	WR

Risikonr	Risiko	Beschreibung	Eintrittswahrscheinlichkeit [%]	Schadenspotenzial[h]	Reserven [h]	Vermeidungs- und Verminderungsmassnahmen	Aktionen beim Eintreffen	Massnahmen nur beim Eintreffen	Betroffene Arbeitspakete oder Massnahmen	Spätester Zeitpunkt der Risikobeseitigung (Iteration)	Risiko bereinigt
R01	SVN Repository Ausfall	Der SVN Server, welcher von der HSR bereitgestellt wird, fällt aus. Dadurch ist der Zugriff auf Dokumente nicht mehr möglich	0	8	0	SVN regelmässig auschecken um lokale Kopien zu haben. Die meisten Dokumente werden direkt auf dem Wiki des Redmine-Portals erstellt und bearbeitet, dadurch wäre ein Ausfall weniger schwerwiegend.	Sobald SVN wieder verfügbar ist, werden Daten mit dem Server abgeglichen.	-	Alle	Bis Ende Projekt	X
R02	Ausfall Redmine	Der von der HSR bereitgestellte virtuelle Server, steigt aus und damit wird der Zugriff auf das zentrale Medium für das Projekt verunmöglicht.	0	8	0	Da der Unterhalt des Servers von dafür zuständigen Personen durchgeführt wird, haben wir keine Möglichkeiten darauf einfluss zu nehmen.	Die Zeiterfassung wird mittels einem temporären Excel-File solange überbrückt, bis der Server wieder erreichbar ist. Um Dokumente zu erstellen bzw. zu bearbeiten wird auf die alternative Google Docs ausgewiechen.	-	Alle	Bis Ende Projekt	X
R03	Zeit während WK zu knapp	Die für die Studienarbeit verfügbare Zeit während dem WK ist zu knapp berechnet um alle gewünschten	50	0	0	Während dem Wiederholungskurs soll die verfügbare Zeit möglichst effizient genutzt werden. Zudem soll durch die Planung beim Beginn des WK, klar sein welche Ziele bis am Ende erreicht werden sollen.	Der Projektplan müsste überarbeitet werden, indem man einzelne Arbeitspakete später durchführt. Eventuell müsste die Arbeitszeit für die restliche Projektzeit	1. Projektplan überarbeiten 2. Arbeitspakete verschieben 3.	Projektplan wurde überarbeitet, Woche beantragt	Bei Beginn Elaboration	X

		Arbeitspakete zu erledigen					erhöht werden.	Zusätzliche Woche beantragen			
R04	Unstimmigkeiten im Team	Es kommt vermehrt zu grösseren Konflikten, welche zu einem massiven Anstieg an Arbeitzeit führt.	0	70	0	Durch viele Team-Höcks und Absprachen wird versucht ein kollegiales Klima zu schaffen. Eventuell treffen wir uns auch regelmässig	Bei Aufkommen von unstimmigkeiten und Problemen inerhalb des Teams, soll dies möglichst offen und direkt mit der entsprechenden Person besprochen werden.	-	-	Bis Ende Projekt	X
R05	Remo Waltenspül kann an WK nicht teilnehmen	Remo Waltenspül kann leider am Wiederholungskurs nicht teilnehmen, da das Gesuch zu kurzfristig eingereicht wird.	0	30	0	Es sollen alle Optionen ergriffen werden, damit Remo Waltenspül teilnehmen kann. (Anrufen bei zuständiger Stelle der Armee)	Remo arbeitet von zuhause aus, und ist per Skype bzw. Redmine mit seinen Projektmitgliedern verbunden. Eventuell reist er in dringenden Situationen persönlich an	-	-	Bei Beginn Elaboration	X
R06	Zeitaufwand falsch kalkuliert	Aufgrund der falschen Einschätzung stimmt der Zeitplan nicht	0	60	0	Ständige Überwachung und Projektmonitoring durch bestimmtes Projektmitglied.	Fokus auf die Hauptfunktionalitäten setzen, falls Zeit knapp wird Arbeitspakete mit tiefer Priorität streichen	-	-	Bis Ende Projekt	X

R07	Erhöhte Einarbeitungszeit	Einarbeitungszeit für die relevanten Technologien ist grösser als erwartet	0	80	0	Sich ständig absprechen, bei unklarheiten nach einer bestimmten Zeit Projektmitglieder fragen	Projektplan überarbeiten, unter Umständen den Funktionsumfang anpassen	-	Alle Implementationspakete	Bis Ende Sprint 2	X
					0					
			Total:	256	0					